INTERNE BERICHTE

Carl von Ossietzky Universität Oldenburg

Fachbereich Informatik

Zwischenbericht der Projektgruppe

Werkzeuge für Digitale Bibliotheken

J. Eschke, M. Höft, M. Hülsmann, M. Klein, O. Klein, M. Malachinski, T. Ottenhues, D. Pawlowski, A. Rolfs, V. Weber, W. Willms, A. Wunsch, D. Boles, C. Haber

Bericht IS xx

Teil A

Oktober 2000

Zusammenfassung

Dieser Bericht faßt die Ergebnisse zusammen, die etwa bei *Halbzeit* der Projektgruppe tooLib (Werkzeuge für Digitale Bibliotheken) vorliegen. tooLib ist eine Projektgruppe des Fachbereichs Informatik der Universität Oldenburg und findet im Wintersemester 1999/2000 und im Sommersemester 2000 statt.

Dieser Zwischenbericht setzt sich aus zwei Teilen zusammen:

- **Teil A** faßt den bisherigen Ablauf und die bisherigen Ergebnisse der Projektgruppe zusammen.
- **Teil B** enthält die ausgearbeiteten Seminarvorträge aus dem ersten Abschnitt der Projektgruppe.

Nach Abschluß der Projektgruppe im Sommer 2000 wird ein Endbericht – ebenfalls als Interner Bericht – erscheinen.

Inhaltsverzeichnis

1 Rahmenbedingungen **1**

1.1 Was ist eine Projektgruppe? 1

1.1.1 Definition von Projektgruppen 1

1.1.2 Erläuterungen zum Zweck von Projektgruppen 1

1.1.3 Hinweise für Veranstalter und Studierende 2

1.2 Projektgruppenantrag 3

1.2.1 Formalia 3

1.2.2 Aufgabenstellung 4

1.2.3 Literatur 6

1.3 Teilnehmer 6

2 Seminarphase **9**

2.1 Objektorientierte Softwareentwicklung mit dem Rational Unified Process 9

2.2 Objektorientierte Analyse mit der UML und Use Cases 9

2.3 Objektorientierter Entwurf mit Frameworks und Entwurfsmustern 10

2.4 Objektorientierte Programmierung mit Softwarekomponenten 10

2.5 Client/Server-Programmierung mit Java 10

2.6 Datenbankanbindung für Informationssysteme 11

2.7 Traditionelles und digitales Publikations- und Bibliothekswesen 11

2.8 Digitale Bibliotheken im Vergleich 11

2.9 Information Retrieval 12

2.10 Metadaten 12

2.11 XML 12

2.12 Elektronischer Zahlungsverkehr im Internet 13

3 Ergebnisse der Bean Digital Library **15**

3.1 Einleitung 16

3.2 Meilensteinplan 17

3.3 Entwurf und Entwicklung 18

3.3.1 NewBeanBox 18

3.3.2 Server 21

3.3.3 Das BeanDL - Webinterface 21

3.3.4 Use-Cases 24

3.4 Designentscheidungen und Art der Implementierung 32

3.5 Benutzerhandbuch 33

3.5.1 NewBeanBox 33

3.5.2 Installation und Inbetriebnahme des BeanServers 40

3.5.3 Das BeanDL - Webinterface 41

3.6 Abschließende Anmerkungen 47

4 Ergebnisse der Framework-Gruppe **49**

4.1 Aufgaben-Beschreibung 50

4.1.1 Die gestellte Aufgabe 50

4.1.2 Vorgehensweise 50

4.2 Analyse 51

4.2.1 Anforderungsdefinition 51

4.2.2 Entitätenbestimmung 54

4.3 Entwurf 54

4.3.1 Designentscheidungen 54

4.3.2 Entwurf des Datenmodells 55

4.3.3 Entwurf der Benutzerschnittstelle 60

4.4 Implementierung 70

4.4.1 Schnittstellendokumentation 70

4.4.2 Benutzerhandbuch des Käufers 84

4.4.3 Benutzerhandbuch der Verkäufers 88

5 Ergebnisse der Sport Digital Library **97**

5.1 Einleitung . . . 98

5.1.1 Motivation . . . 98

5.1.2 Das Sub-Projekt Sport-DL . . . 98

5.1.3 Aufbau . . . 98

5.2 Zusammenfassung der Anforderungsdefinition . . . 99

5.2.1 Die Dokumenttypen des Informationssystems . . . 99

5.2.2 Einstellungsverfahren . . . 99

5.2.3 Funktionen im Informationssystem . . . 99

5.2.4 Weitere Datenbestände . . . 100

5.2.5 Evaluation . . . 100

5.3 Analyse . . . 100

5.3.1 Entwicklungsphasen mit Meilensteinen . . . 100

5.3.2 Anwendungfälle . . . 101

5.3.3 Use Case Diagramm mit den Anwendungsfällen des Nutzers . . . 108

5.4 Design . . . 108

5.4.1 Das Datenmodell zur Sport-DL . . . 108

5.4.2 Definition der Benutzerschnittstellen . . . 110

5.5 Evaluation . . . 117

5.5.1 Komponenten, die nachher verwendet wurden . . . 117

5.5.2 Komponenten, die vorausgesetzt werden mußten . . . 118

5.5.3 Komponenten, die keiner größeren Evaluation unterworfen wurden . . . 118

5.5.4 Komponenten, die einer vollständigen Evaluierung unterworfen wurden . 119

5.5.5 Komponenten, die unabhängig vom Projekt waren . . . 119

5.6 Implementierung und Test . . . 120

5.6.1 Einsatz zusätzlicher Tools und Komponenten . . . 120

5.6.2 Implementierungsaspekte . . . 120

5.6.3 Beschreibung der Hilfsklassen . . . 124

5.7 Handbuch . . . 126

5.7.1 Suche und eJournal . . . 126

5.7.2 Suchergebnisse . . . 126

5.7.3 Konfiguration ändern . . . 126

5.7.4 Registration 127
5.7.5 Die Erweiterte Suche 127
5.7.6 Dokumente einspielen 127
5.7.7 Kommunikation 128

Abbildungsverzeichnis

3.1 Die neue ToolBox 34
3.2 Das PopUp Menü (lokale JavaBean) 35
3.3 Das PopUp Menü (remote JavaBean) 36
3.4 Die Konfigurationseinstellungen 37
3.5 Die Suchmaske 38
3.6 Die Suchergebnisse 39
3.7 Die Registrierung einer JavaBean 42
3.8 Der Update einer JavaBean 44
3.9 Das Löschen einer JavaBean 45

4.1 Das Framework-Datenbankschema 62
4.2 Login 63
4.3 Ist eingeloggt 63
4.4 Benutzer anlegen 64
4.5 Benutzer ändern 64
4.6 Konto einsehen 64
4.7 Angebote ansehen 65
4.8 Angebot suchen 65
4.9 Angebot auswaehlen 65
4.10 Angebot bestellen 66
4.11 Lizenzen anzeigen 66
4.12 Bezahlen 66
4.13 Ware eingeben 67
4.14 Ware ändern 67
4.15 Bundle anlegen 67

4.16 Angebot anlegen . . . 68
4.17 Warengruppe löschen . . . 68
4.18 Benutzer Anlegen . . . 69
4.19 Kontostand anzeigen . . . 69
4.20 Waren ansehen . . . 69
4.21 Warenkorb ansehen . . . 69
4.22 Angebote suchen . . . 69
4.23 Suchergebnisse anzeigen . . . 69
4.24 Lizenzen anzeigen . . . 70
4.25 Angebote auswählen . . . 70
4.26 Waren eingeben . . . 70
4.27 Warengruppe anlegen . . . 70
4.28 Waren zu Bundle zuordnen . . . 70
4.29 Bundle anlegen . . . 70
4.30 Angebot anlegen . . . 71
4.31 WarengruppeLoeschen . . . 71
4.32 WarengruppeLoeschenDialog . . . 71
4.33 Anlegen eines neuen Benutzers . . . 84
4.34 Anlegen einer Käufergruppe . . . 85
4.35 Auswählen von Waren . . . 86
4.36 Auswählen von Angeboten . . . 86
4.37 Ansicht des Warenkorbs . . . 87
4.38 Dialog zum Bestellen von Angeboten . . . 87
4.39 Anzeige der gültigen Lizenzen des Käufers . . . 88
4.40 Anzeige des Kontostandes . . . 88
4.41 Anlegen einer Warengruppe . . . 90
4.42 Anlegen einer Ware . . . 91
4.43 Anlegen eines Lizenzmodells . . . 92
4.44 Zuordnen von Lizenzmodellen zu Waren . . . 93
4.45 Erstellen von neuen Angeboten . . . 94
4.46 Hinzufügen von Käufern zu Gruppen . . . 94
4.47 Anlegen einer Käufergruppe (Verkäufer) . . . 95

5.1 Use Case Diagramm mit den Anwendungsfällen des Nutzers 108
5.2 Das Datenmodell zur Sport-DL 109
5.3 Die Startseite der Sport-DL 110
5.4 Die Registration 110
5.5 Login 111
5.6 Startseite nach dem Einloggen mit Möglichkeiten der Suche 111
5.7 Erweiterte Suche 112
5.8 Beispiel zu den Suchergebnissen 113
5.9 Dokumente einspielen 113
5.10 Volltext einspielen 114
5.11 Abstract zu einem speziellen Volltext nachträglich einspielen 115
5.12 Eingabe der Daten zu einem zusätzlichen Abstract 115
5.13 Einspielen von sonstigen Dokumenten 116
5.14 Kommunikation in der Sport-DL 116
5.15 Konfiguration der Benutzereinstellungen 117

Kapitel 1

Rahmenbedingungen

1.1 Was ist eine Projektgruppe?

Im folgenden wird das Selbstverständnis dieser Lehrveranstaltungsform in Form einer Definition, Erläuterungen zum Zweck sowie Hinweise für Veranstalter und Studierende einer Projektgruppe beschrieben. Dieses Papier lag bereits zum Jahreswechsel 87/88 im wesentlichen fest und wurde nach kleinen Änderungen am 1.6.1988 von der Studienkommission des Fachbereichs Informatik der Universität Oldenburg verabschiedet.

1.1.1 Definition von Projektgruppen

Projektgruppen sollen die Besonderheiten eines Fortgeschrittenenpraktikums, eines Seminars und einer Studienarbeit vereinen und zugleich berufstypische Arbeitsweisen im Studium vermitteln. Eine Projektgruppe besteht aus acht bis zwölf Studierenden, die ein Jahr lang ein umfangreiches Problem bearbeiten. Insgesamt wird diese Lehrveranstaltung mit 15 Semesterwochenstunden angesetzt. Die Themen für Projektgruppen können sowohl aus der *Kern-Informatik* als auch aus ihren Anwendungsgebieten stammen. Es ist möglich (und auch sinnvoll), daß sich an einer Projektgruppe Wissenschaftler anderer Fachrichtungen oder externe Kooperationspartner beteiligen. Eine Projektgruppe sollte i. allg. durch eine Stammvorlesung und/oder eine Spezialvorlesung vorbereitet werden. In der Regel werden aus einer Projektgruppe mehrere Diplomarbeiten hervorgehen.

1.1.2 Erläuterungen zum Zweck von Projektgruppen

Zu vermittelnde berufstypische Arbeitsweisen sind:

- Arbeiten im Team (insbesondere: Präzisierung und Definition von Schnittstellen, Aufgabenverteilung, Zuständigkeiten und Verläßlichkeiten in einer Gruppe)
- Umfassendere Betrachtungsweisen bei der Software-Entwicklung (Kennenlernen des gesamten Software-Lebenszyklusses, Verwendung unterschiedlicher Spezifikationssprachen, Ist- und Soll-Analysen, Kosten-Nutzen-Analysen, Einsatz- und Auswirkungsproblematik, Standard- und kundenspezifische Software)

- Einsatz von Werkzeugen (Sichtung vorhandener Software, Planung und Auswahl von Sprachen, Nutzung von Software-Entwicklungswerkzeugen, maschinenabhängige und maschinenunabhängige Konzepte)
- Konkrete Erstellung von Software unter gleichzeitiger Anfertigung einer Dokumentation und weiterer Materialien (Handbücher, Konfigurierungen, Wartungsanweisungen usw.)
- Arbeitsstil und persönliche Befähigungen (Organisation umfangreicher Projekte, Präsentation von Resultaten und Teilnahme an Diskussionen, Sitzungen mit präziser Protokollierung, Planungs- und Konfliktmanagement, Einarbeitung und Beschaffung von Literatur, Einblick in arbeitspsychologische Phänomene)

1.1.3 Hinweise für Veranstalter und Studierende

Entsprechend dieser Ziele sollte eine Projektgruppe nach folgendem Schema geplant werden und ablaufen:

1. Die Vorgaben durch die Veranstalter umfassen:
   - Grobziele der Projektgruppe mit Themenstellung und Zielsetzungen
   - Zeitplanung über zwei Semester
   - Literaturangaben und Vorträge für die Seminarphase
   - Vorschläge für Selbstverwaltungsaufgaben
   - Teilnahmevoraussetzungen
   - präzise Kriterien für die Erlangung des Projektgruppenscheins
   - benötigte Ressourcen des Fachbereichs und Entwicklungsumgebungen
2. Minimalergebnisse:
   Von jedem Teilnehmer wird die aktive Mitarbeit an den Projektgruppensitzungen, die Vorbereitung und Abhaltung eines oder zweier Seminarvorträge, die Erfüllung von bestimmten Verwaltungsaufgaben innerhalb der Projektgruppe, die Mitarbeit an einem Zwischen- und einem Endbericht und die Erfüllung der übertragenen Programmieraufgaben erwartet. Frühzeitige Aussprachen über mangelnde Mitarbeit oder Desinteresse bei Beteiligten gehören zum Konfliktmanagement und sollten keinesfalls verdrängt werden. Von besonderer Bedeutung ist die kontinuierliche Erarbeitung einer Dokumentation, an der alle Teilnehmer in gleichem Maße mitwirken müssen. Dabei geht es nicht um ein Handbuch (das ebenfalls erstellt werden muß), sondern um die systematische Darstellung z. B. folgender Bereiche:
   - Problemstellung, Literatur, Überblick
   - Ist-Analyse und Soll-Konzept
   - Anforderungen, Spezifizierungen, Benutzungsschnittstellen
   - Empirische und formale Evaluation
   - Modulbeschreibungen und Implementierung
   - Integration und Tests

- Wartung, Fehlerfälle, Portierungsmöglichkeiten
- Erweiterungen, Ausblick.

3. Die Zeitplanung kann nach folgendem Schema ablaufen:

   - Vorlesungsfreie Zeit vor der Projektgruppe:
     Vorbereitung der Vorträge der 1. Seminarphase
     Einarbeitung in die Entwicklungsumgebungen
   - Erstes Semester:
     Seminarphase (ca. vier Wochen)
     Planungs- und Entwurfsphase (ca. vier Wochen)
     Spezifikationen, Aufgabenverteilung, Probeimplementierungen (ca. vier Wochen)
   - Vorlesungsfreie Zeit:
     Beginn der Implementierungsphase
   - Zweites Semester:
     Weiterführung der Implementierungsphase
     Integrationsphase und Tests (ca. vier Wochen)
     Seminarphase
     Erprobung, Handbuch, Dokumentation
   - Vorlesungsfreie Zeit nach der Projektgruppe:
     Präsentation der Projektgruppenergebnisse im Fachbereich Informatik.

Empfehlenswert ist eine Exkursion zu Firmen/Institutionen, die sich mit ähnlichen Fragen wie die Projektgruppe beschäftigt.

Hinweis: Projektgruppen sind von den Gremien des Fachbereichs Informatik zu genehmigen. Hierzu ist zunächst das Thema und die Ankündigung (einschl. der unter 1 bis 3 genannten Punkte) im vorangehenden Semester der Studienkommission zur Bestätigung vorzulegen. Da eine Projektgruppe ein Seminar, ein Fortgeschrittenenpraktikum und eine Studienarbeit ersetzt, muß der Diplom-Prüfungsausschuß die Äquivalenz der Projektgruppe mit diesen Studienleistungen genehmigen. Anschließend kann die Projektgruppe ausgeschrieben werden. Eine Projektgruppe soll nicht begonnen werden, wenn weniger als acht Studierende an ihr teilnehmen.

1.2 Projektgruppenantrag

Die nachfolgende Übersicht faßt den im Fachbereich gestellten Antrag der Veranstalter auf Durchführung der Projektgruppe in seinen formalen Angaben und der Aufgabenstellung zusammen.

1.2.1 Formalia

1.2.1.1 Veranstalter

Dipl.-Inform. Dietrich Boles
Dipl.-Inform. Cornelia Haber

1.2.1.2 Zeitraum

WS 99/00 und SS 00

1.2.1.3 Umfang

beide Semester je 8 SWS

1.2.1.4 Lehrveranstaltungsäquivalent

1 Seminar
1 Fortgeschrittenenpraktikum
1 Studienarbeit

1.2.1.5 Inanspruchnahme von Fachbereichsressourcen

Der Rechner- und Softwarebedarf wird durch Ressourcen der veranstaltenden Abteilung befriedigt. Ein Raum für Sitzungen steht im OFFIS-Gebäude zur Verfügung.

1.2.1.6 Teilnahmevoraussetzungen

Abgeschlossenes Grundstudium mit erfolgreich abgeschlossenem Vordiplom zu Beginn des WS 99/00.

1.2.2 Aufgabenstellung

1.2.2.1 Zielsetzung

Digitale Bibliotheken sind verteilte Informationssysteme, in denen Dokumente in elektronischer Form gespeichert werden, die die Dokumente verwalten und die Zugriff auf die Dokumente gewähren. Die gespeicherten Dokumente können dabei rein textueller Natur sein, aber auch multimediale Daten enthalten. Digitale Bibliotheken implizieren eine Reihe von Vorteilen gegenüber traditionellen Bibliotheken wie einen schnellen Zugriff via Internet, eine Volltextrecherche oder eine gegenseitige Verbindung der einzelnen Dokumente über Hyperlinks.

In einer digitalen Bibliothek stellen Anbieter wie Verlage, Bibliotheken oder Autoren potentiellen Lesern die Dokumente zur Verfügung. Digitale Bibliotheken können also als Informationsvermittlungssysteme zwischen Informationsproduzenten bzw. -anbietern und Informationskonsumenten betrachtet werden. Funktionale Anforderungen an eine digitale Bibliothek bzgl. der Produzenten bzw. Anbieter sind etwa die Integration neuer Dokumente oder die Abrechnung kostenpflichtiger Dokumente. Konsumenten müssen insbesondere flexible Zugriffsmöglichkeiten wie eine Suche über bibliographische Attribute, eine Volltextsuche, die Navigation im Dokumentenbestand und Profildienste angeboten werden. Des weiteren sind auch die Betreiber einer digitalen Bibliothek durch entsprechende Dienste wie die Aufstellung von Nutzungsstatistiken oder die Einbindung neuer Datenbanken in das verteilte System zu unterstützen.

Ziel der Projektgruppe ist die Konzeption und Realisierung eines Werkzeugkastens für die Entwicklung verteilter digitaler Bibliotheken im Internet. In einer ersten Phase werden dazu existierende tradionelle und digitale Bibliotheken bzgl. Struktur, Funktionalität, Arbeitsweise und Datenfluß untersucht. Die Ergebnisse der Analyse fließen ein in die Entwicklung eines Referenzmodells, das den generellen Aufbau und die generelle Funktionalität digitaler Bibliotheken beschreibt. Als Formulierungsnotation wird hierzu die Unified Modeling Language (UML) verwendet. Einzelne Komponenten dieses Referenzmodells werden anschließend in Form eines generischen Frameworks in Java implementiert. Zur Evaluation des Referenzmodells sowie des Frameworks wird abschließend eine konkrete digitale Bibliothek aufgebaut.

1.2.2.2 Entwicklungsumgebung

- Hardware:
  - SUN-Workstations
  - PCs unter WindowsNT
  - Apple Macintosh Performas 5200
- Software:
  - Netscape Navigator
  - Microsoft Explorer
  - Java Developers Kit 1.1 (JDK)
  - Bean Developers Kit (BDK)
  - C++-Compiler
  - Rational Rose (UML)
  - Perl (zur Realisierung von CGI-Skripten)
  - LaTeX (für die Dokumentation)

1.2.2.3 Minimalergebnisse

- Aktive Mitarbeit bei der Analyse, Konzeption und Implementierung
- Erfüllung übertragender Aufgaben
- Präsentation und Ausarbeitung von Seminarvorträgen
- Erstellung von Zwischen- und Endbericht sowie anfallenden Dokumentationen

1.2.2.4 Zeitplanung

- WS 99/00:
  - Seminarphase
  - Einarbeitung in Internet-Technologien und -entwicklungswerkzeuge
  - Analyse existierender tradioneller und digitaler Bibliotheken

  - Entwicklungs eines Referenzmodells für digitale Bibliotheken
  - Konzeption eines Frameworks für digitale Bibliotheken

- Vorlesungsfreie Zeit nach Ende des WS 99/00:
  - Anfertigung des Zwischenberichts
  - Implementierung einzelner Komponenten des Frameworks

- SS 00:
  - Konzeption einer konkreten digitalen Bibliothek mit Hilfe des Referenzmodells und des Frameworks
  - Implementierung der digitalen Bibliothek

- Vorlesungsfreie Zeit nach Ende des SS 00:
  - Anfertigung des Endberichts
  - Fertigstellung der Dokumentation
  - Abschlußpräsentation

1.2.3 Literatur

Hinweise zu Fachliteratur im Bereich "Internet-Technologien" sowie eine Vielzahl von Referenzen auf Online-verfügbare Informationen zu diesen Themenbereichen finden sich unter:

- http://www-is.informatik.uni-oldenburg.de/~dibo/links/internet.html
- http://www-is.informatik.uni-oldenburg.de/~dibo/teaching/java/links/
- http://www-is.informatik.uni-oldenburg.de/~dibo/teaching/mm/links/

1.3 Teilnehmer

Die Projektgruppe setzt sich aus folgenden zwölf Studierenden zusammen:

- Jens Eschke
- Maik Höft
- Michael Hülsmann
- Michael Klein
- Oliver Klein
- Michael Malachinski
- Tobias Ottenhues

- Daniel Pawlowski
- Andre Rolfs
- Volker Weber
- Werner Willms
- Axel Wunsch

Veranstalter sind Dipl.-Inform. Dietrich Boles und Dipl.-Inform. Cornelia Haber.

Kapitel 2

Seminarphase

Die erste Seminarphase der Projektgruppe diente dazu, den Kenntnisstand der Teilnehmer zum einen bezüglich moderner Methoden des Software-Engineering zum anderen bezüglich der eigentlichen Thematik der Projektgruppe – Digitale Bibliotheken – zu verbessern und zu vereinheitlichen.

Dieses Kapitel enthält Zusammenfassungen der einzelnen Vorträge. Die vollständigen Ausarbeitungen finden sich im Teil B des Zwischenberichts.

2.1 Objektorientierte Softwareentwicklung mit dem Rational Unified Process

Der Rational Unified Process ist ein objektorientierter Software-Entwicklungsprozess, der von der Firma Rational entwickelt wurde. Der Software-Entwicklungsprozess wird in vier Phasen und neun logische Aufgaben unterteilt. Er ist als ein iterativer Prozess ausgelegt, was dazu führt, dass schon in frühen Entwicklungsstadien Prototypen entwickelt sind und so Risiken und Probleme auch in frühen Stadien erkannt werden können. Der Rational Unified Process definiert Workflows in denen beschrieben wird, welche Aufgaben zu welchem Zeitpunkt durchzuführen sind.

Dieser Artikel gibt eine Übersicht über die wichtigsten Eigenschaften und Elemente des Rational Unified Process. *Volker Weber*

2.2 Objektorientierte Analyse mit der UML und Use Cases

Software wird immer komplexer, und die Entwickler benötigen Hilfsmittel und Werkzeuge, die den steigenden Anforderungen gerade für den objektorientierten Bereich gerecht werden.

In dieser Ausarbeitung wird eine kurze Einführung in die objektorientierte Analyse gegeben und die wesentlichen Elemente der Unified Modeling Language anhand deren Notation und Semantik erklärt. Außerdem wird hier der Ansatz einer Analysemethode vorgestellt, die sich hauptsächlich an Use Cases (Anwendungsfällen) orientiert. Es wird gezeigt, wie ein Softwareentwickler mit Hilfe von Use Cases von einer externen Anwendersicht immer weiter zu einer komplexen inneren

Sichtweise eines Systems gelangen kann, indem er den Use Case als Grundlage für den Einsatz weiterer Modellelemente der UML verwendet.

Oliver Klein

2.3 Objektorientierter Entwurf mit Frameworks und Entwurfsmustern

In diesem Artikel wird das Konzept der Entwurfsmuster und der Frameworks vorgestellt und mittels ausgewählter Beispiele verdeutlicht. Es werden Möglichkeiten aufgezeigt, Entwurfsmuster zu beschreiben, zu klassifizieren und sinnvoll anzuwenden. Auch Frameworks werden hinsichtlich verschiedener Kriterien geordnet. Darüber hinaus werden Vor- und Nachteile der Frameworkverwendung, sowie Probleme bei der Frameworkentwicklung aufgeführt.

Maik Höft

2.4 Objektorientierte Programmierung mit Softwarekomponenten

Komponenten sind wiederverwendbare Softwarefragmente, die mit anderen Komponenten einfach zu neuen Anwendungen kombiniert werden können. Diese Seminarausarbeitung beschäftigt sich mit dem Thema der Entwicklung von Softwarekomponenten. Es wird zunächst erklärt, was Softwarekomponenten sind und wo sie Verwendung finden. Dann wird ausführlich auf die Komponententechnologie JavaBeans von JavaSoft eingegangen. Die Struktur einer JavaBean und die Kommunikation zwischen Beans sind wesentliche Punkte bei der Beschreibung von JavaBeans. Schließlich wird die Anwendung von Beans an einem Beispiel gezeigt. Abschließend werden dann noch einige Entwicklungswerkzeuge vorgestellt und ein Fazit gezogen.

Axel Wunsch

2.5 Client/Server-Programmierung mit Java

Die Client/Server-Programmierung ist das Standard-Programmiermodell für vernetzte Rechner: ein Client sendet eine Anfrage an einen entfernten Server, der bearbeitet die Anfrage und sendet eine Antwort zurück.

Diese Seminarausarbeitung beschäftigt sich mit der Entwicklung von verteilten Applikationen, die in der Programmiersprache Java geschrieben werden. Im Wesentlichen wird dabei auf Sockets, RMI und CORBA eingegangen. Die Grundlagen dieser Konzepte werden kurz dargestellt. Als Ausgangspunkt für diese Konzepte dient die Client/Server-Architektur, als das Progammiermodell für verteilte Anwendungen in Netzwerken. Um einen praktischen Bezug zu erhalten, wird anhand von Beispielen die Umsetzung dieser Konzepte deutlich gemacht. Die Beispiele sind einfach gehalten, um ein Nachvollziehen der Vorgehensweise zu erleichtern.

Michael Hülsmann

2.6 Datenbankanbindung für Informationssysteme

Diese Ausarbeitung soll einen Überblick über Internet-basierte Datenbank-Anbindungen geben. Es wird erläutert, was eine Datenbank ist, wie man sie generell ansteuert und wie man sie über das Internet zugreifbar macht. Dazu werden unterschiedliche Techniken vorgestellt, die sich zwar nicht rein auf die Datenbank-Anbindung beziehen, aber oft dafür benutzt werden. Dazu gehören CGI, ASP, Java-Applets und -Servlets und speziell ODBC und JDBC, wobei die zwei zuletzt genannten Techniken ausschließlich auf die Anbindung von Datenbanken ausgerichtet sind. Zum Schluss wird auf die Anbindung von Datenbanken über proprietäre Schnittstellen im Gegensatz zu freien Schnittstellen eingegangen.

Andre Rolfs

2.7 Traditionelles und digitales Publikations- und Bibliothekswesen

Vor 5000 Jahren entwickelten sich die ersten Schriften. Und mit ihnen wenig später die ersten Bibliotheken. Die Bibliotheken blieben aber nicht immer in der gleichen Form bestehen, sondern passten sich in einer evolutionären Entwicklung stets der kulturellen Umgebung an. So entwickelte sich im Laufe der Jahrhunderte die Bibliothek, wie wir sie heute kennen. Doch auch sie wird sich weiterentwickeln müssen, da sie in unserer Zeit immer mehr an ihre Grenzen stößt. Erste Ansätze dazu wurden bereits in Form von digitalen Bibliotheken gemacht.

Aufgabe dieser Seminar-Arbeit ist es zu zeigen, welche Änderungen die digitalen Bibliotheken im Publikationsprozeß und Bibliothekswesen mit sich bringen werden.

Daniel Pawlowski

2.8 Digitale Bibliotheken im Vergleich

Dieser Bericht befasst sich mit digitalen Bibliotheken, und dem Vergleich einiger ausgewählter digitaler Bibliotheken, die einen näheren Bezug zum Fachgebiet Informatik haben. Die Einleitung beschreibt welche Vorteile sich sowohl für die Anbieter als auch für die Benutzer bieten, wenn sie digitale Bibliotheken, anstatt herkömmlicher Bibliotheken, benutzen. Danach folgt die Einteilung der digitalen Bibliotheken in Kategorien, das heißt digitale Bibliotheken die sich ähneln werden unter einer Kategorie zusammengefasst. Als letztes folgt in der Einleitung welche Kriterien sich für einen Vergleich von digitalen Bibliotheken eignen. Mit der Vorstellung der einzelnen digitalen Bibliotheken befasst sich der nächste Abschnitt. Zu den in diesem Bericht vorgestellten digitalen Bibliotheken gehören: ACM, IEEE-CS, LINK, NCSTRL, NZDL, MeDoc und UniCats. Einige der vorgestellten digitalen Bibliotheken beinhalten sehr unterschiedliche Ansätze zur Architektur von digitalen Bibliotheken. In den abschließenden Anmerkungen kann man noch einmal die wichtigsten Unterschiede der vorgestellten digitalen Bibliotheken nachlesen. Als letztes folgt, für die weitere Vertiefung in dieses Gebiet, die für diesen Bericht verwendete Literatur.

Jens Eschke

2.9 Information Retrieval

Diese kurze Ausarbeitung zum Thema Information Retrieval im Rahmen der Projektgruppe "Entwicklung von Werkzeugen für Digitale Bibliotheken" an der CvO-Universität Oldenburg soll die wichtigsten Grundlagen der Informationswiedergewinnung digitaler Daten darstellen. Nach einer kurzen Einleitung wird auf das Information Retrieval bei Texten und die dadurch entstehenden Probleme eingegangen. Anschließend werden grundlegende Ansätze des Information Retrieval aufgezeigt und die heute wichtigsten - unter anderem das Boolesche Retrieval - inklusive deren Vor- und Nachteile erläutert. Abschließend wird eine Bewertung gegenwärtiger Systeme vorgenommen, sowie ein Ausblick auf zukünftige Entwicklungen gewährt.

Werner Willms

2.10 Metadaten

Metadaten stellen Informationen über Dokumente zur Verfügung, mit deren Hilfe diese Dokumente identifiziert und auffindbar gemacht werden sollen. Dies ist gerade in Bezug auf Dokumente, die für eine breite Öffentlichkeit im Internet zugänglich sind, von großer Wichtigkeit. Mit Metadaten soll die Katalogisierung von Publikationen effizient und effektiv erreicht werden, um so ein gezieltes Auffinden von relevanten Informationen zu ermöglichen.

Dieses Dokument gibt eine Motivation der Einführung von strukturierten Metadaten. Über die allgemeine Definition und Erläuterung von Metadaten hinaus, wird insbesondere auf den *Dublin Core Element Set (DC)* der Dublin Core Metadata Initiative sowie das *Resource Description Framework (RDF)* des W3 Consortiums näher eingegangen.

Michael Klein

2.11 XML

Diese Ausarbeitung zum Thema XML und dem Bezug von XML zu Java wurde im Rahmen der Seminar-Phase der Projekt-Gruppe "Entwicklung von Werkzeugen für digitale Bibliotheken angefertigt.
In den ersten Kapiteln wird eine kurze allgemeine Einführung in XML gegeben und insbesondere auf Teilaspekte wie "DTD", "DOM", "CSS und XSL" näher eingegangen. Anhand eines Beispiels wird die Thematik etwas näher gebracht. Anschließend widmet sich ein Kapitel MathML, einer Anwendung von XML. Der zweite Teil beschäftigt sich dann mit dem Bezug von XML zu Java. Nach dem Vorstellen von Gemeinsamkeiten wird auf den Verarbeitungszyklus eines XML-Dokuments eingegangen. Anschließend werden ein paar java-basierte Werkzeuge für XML vorgestellt und das Kapitel mit einem anschaulichen Beispiel abgeschlossen. In der Literatur- & Link-Liste wird weiterführende Literatur zu den Themengebieten aus dem WWW angeboten.

Tobias Ottenhues

2.12 Elektronischer Zahlungsverkehr im Internet

Diese Ausarbeitung befasst sich mit Zahlungsweisen im Internet. Der Schwerpunkt liegt dabei auf den Verfahren, die eine Zahlung ohne Medienbruch erlauben (im Gegensatz zu den konventionellen Verfahren wie Rechnung oder Nachnahme).
Zuerst werden sowohl die sicherheitstechnischen Anforderungen als auch die des Benutzers an solche Systeme erläutert. Für das bessere Verständnis der Umsetzung der Sicherheitsanforderungen wird auf die Grundlagen kryptographischer Verfahren eingegangen. Es folgt eine Klassifikation von Zahlungsverfahren nach unterschiedlichen Gesichtspunkten, in der auch die grundsätzlichen Konzepte elektronischer Zahlungsverfahren erläutert werden. Im Anschluss werden drei konkrete Zahlungsverfahren untersucht. Abschließend wird ein Blick in die Zukunft und auf die eventuelle Standardisierung elektronischer Zahlungsverfahren geworfen.

Michael Malachinski

Kapitel 3

Ergebnisse der Bean Digital Library

Jens Eschke,
Michael Hülsmann,
Tobias Ottenhues,
Volker Weber

Dieser Bericht befasst sich mit den Ergebnissen der Gruppe Bean Digital Library, kurz BeanDL, deren Aufgabe es war, die BeanBox von SUN zu einer verteilten digitalen Bibliothek zu erweitern. Der danach folgende Meilensteinplan gibt an, in welchem zeitlichen Rahmen die einzelnen Aufgaben bearbeitet werden mussten. Der Abschnitt Entwurf und Entwicklung zeigt wie die NewBeanBox, der Server und das BeanDL-Webinterface entworfen wurden, und welche Entwicklungen die einzelnen Komponenten durchlaufen haben. Im Folgenden werden die Use-Cases, die nach dem Entwurf erstellt wurden, aufgeführt. Anschließend wird kurz auf die Designentscheidungen eingegangen. Dann folgen die Art der Implementierung und das Benutzerhandbuch mit zugehörigen Screenshots. Zu diesem Benutzerhandbuch gehört vor allem, wie der Benutzer mit der NewBeanbox, dem Server und dem BeanDL-Webinterface arbeiten sollte. Die abschließenden Anmerkungen beenden dann diesen Bericht.

3.1 Einleitung

Die Aufgabe der BeanDL-Gruppe war es, auf Grundlage des Bean Development Kit 1.1 und des Java Development Kit 1.2, eine neue BeanBox zu erstellen, diese neue BeanBox bekam von den Mitgliedern der BeanDL-Gruppe den Namen NewBeanBox. Die NewBeanBox sollte der Aufgabenstellung entsprechend angepasst werden. Diese Aufgabenstellung umfasste vor allem die Umwandlung des BDK in eine verteilte digitale Bibliothek auf Grundlage von JavaBeans. Das heißt das bestehende BDK musste so umgewandelt werden, dass es sowohl JavaBeans aus dem Internet als auch JavaBeans, die lokal auf der Festplatte liegen, verwenden kann. Diese Aufgabe erforderte eine Anpassung der ToolBox, sowie die Implementierung eines Servers, über den die NewBeanBox auf die JavaBeans im Internet Zugang erhält und eine WWW-Schnittstelle für die KleinAnbieter einzelner JavaBeans.

Im BDK 1.1 wurden die JavaBeans in einem speziellen Verzeichnis abgelegt. Mit der NewBeanBox soll es möglich sein, JavaBeans über das Internet von verschiedenen Servern zu verwenden aber auch weiterhin lokale JavaBeans zu benutzen.

Eine der weiteren Aufgaben war die Implementierung einer Suche auf den angebotenen JavaBeans. Das heißt man kann die JavaBeans die einem lokal oder über einen der Server zur Verfügung stehen, nach verschiedenen Kriterien durchsuchen.

Bei der Aufgabenstellung war auch vorgesehen die Grundfunktionalitäten der BeanBox nicht zu verändern.

Der Schwerpunkt dieser Gruppe lag auf der Benutzerseite nicht auf der Anbieterseite, das heißt die NewBeanBox wird hauptsächlich für den Anwender optimiert und nur zum geringen Teil für den Anbieter.

3.2 Meilensteinplan

Der erste Meilenstein war der Entwurf der Benutzungsschnittstelle und die Festlegung der Funktionalitäten, die die NewBeanBox leisten soll. Dieser Meilenstein sollte bis zum 21.12.1999 fertiggestellt werden und wurde von der BeanDL-Gruppe termingerecht abgegeben. In diesem Abschnitt traten keine großen Schwierigkeiten auf, denn die Oberfläche der BeanBox war eigentlich schon vorgegeben.

Die Festlegung der Use-Cases war der nächste Meilenstein, dessen Fertigstellung zum 5.1.2000 abgeschlossen sein sollte. Dieser Meilenstein war auch pünktlich fertig, nur der Server musste noch nachträglich angepasst werden. Die Use-Cases waren auch nicht so umfangreich, weil die Funktionalitäten des BDK beibehalten worden sind, und diese nicht extra erwähnt werden mussten.

Bis zum 25.1.2000 sollten der Server und der BeanLoader entworfen sein. Auch hier konnte die BeanDL-Gruppe ihren Termin einhalten, wobei es hier Probleme mit der Art des Servers gab. Die ersten Entwürfe umfassten nur einen Server der sich alle registrierten JavaBeans von den einzelnen Homepages der Anbieter abholen sollte, sofern diese von einem Benutzer der NewBeanBox angefordert wurden. Diese Lösung erwies sich allerdings als zu zentralisiert, denn die Aufgabe war es eine verteilte digitale Bibliothek zu entwerfen. Wäre bei diesem System der Server weggefallen, wäre die komplette digitale Bibliothek nicht mehr verfügbar gewesen. So haben wir uns dafür entschieden, ein System mit mehreren Servern zu entwerfen, wobei ein Server als Master arbeitet. Fällt dieser aus, besteht weiterhin die Möglichkeit sich bei anderen Servern anzumelden und JavaBeans anzufordern.

Die Implementierung sollte dann bis zum 22.2.2000 abgeschlossen sein, so dass man sich an den letzten Meilenstein machen konnte. Und zwar das Testen der NewBeanBox.

Gerade bei der Implementierung tauchten unerwartete Probleme auf; so funktionierte der Juggler anfangs nicht mehr mit der NewBeanBox, weil der SimpleClassLoader des BDK, der von uns so übernommen wurde, einen Fehler aufwies. Außerdem tauchten immer wieder ein paar kleinere Unstimmigkeiten im Programm auf. Deshalb verzögerte sich die Implementierung und die letzten beiden Meilensteine konnten nicht mehr ganz eingehalten werden.

3.3 Entwurf und Entwicklung

Entworfen werden mussten die ToolBox, der Server und das BeanDL-Webinterface, sowie alle Dateien des BDK, die von diesen Änderungen betroffen sind.

Die grundlegenden Änderungen innerhalb des BDK betrafen die ToolBox. Hier wurden bisher lediglich die lokale vorhandenen JavaBeans angezeigt. Die neu zu entwickelnde ToolBox sollte neben den lokal zu erreichenden JavaBeans auch die über Netzwerk von den Servern abrufbaren JavaBeans anzeigen. Zusätzlich sollte eine Suche implementiert werden, die die Möglichkeit bietet, nach bestimmten Kriterien auf den lokalen wie auch auf den entfernten JavaBeans zu suchen.

Da das BDK keine Netzwerkfähigkeiten besitzt, musste ein Server entworfen werden. Über einen Server können Anbieter ihre JavaBeans den NewBeanBox-Benutzern zur Verfügung stellen.

Das BeanDL-Webinterface definiert die Schnittstelle zum KleinAnbieter, der einzelne JavaBeans zur Verfügung stellen möchte.

3.3.1 NewBeanBox

Das Bean Development Kit 1.1 von SUN wird um die folgende Funktionalität erweitert:

In der existierenden BeanBox müssen die JavaBeans in einem speziellen Verzeichnis auf dem eigenen Rechner abgelegt sein. Diese Einschränkung, die Speicherung der JavaBeans auf dem eigenen Rechner, soll durch die NewBeanBox aufgehoben werden. Dem Benutzer soll es ermöglicht werden, JavaBeans von verschiedenen Servern zu verwenden. Um dies zu ermöglichen, wird ein spezieller Server (Meta-Server) eingerichtet, von dem die BeanBox die Informationen über die JavaBeans anfordern kann und diese gegebenenfalls downloaden kann. Zusätzlich hält jeder Server Adressen von anderen Bean-Servern, die ihrerseits wieder JavaBeans enthalten, bereit. Die NewBeanBox bekommt die Adressen dieser Bean-Server geliefert und fordert von diesen ebenfalls Informationen über JavaBeans an bzw. lädt diese JavaBeans herunter. Diese Informationen für eine JavaBean sind:

- Lage der JavaBean im Netz (Die JavaBean befindet sich entweder beim Meta-Server oder bei einem der Bean-Server.)
- Kategorie der JavaBean (Beim Anmelden hat der Anbieter dafür zu sorgen, seine JavaBeans in bestimmte Kategorien einzuordnen. Der sogenannte KleinAnbieter nimmt diese Einordnung beim Anmelden seiner JavaBean auf dem BeanDL-Webinterface des Meta-Servers vor. Anbieter, die den Bean-Server verwenden, müssen dafür Sorge tragen, dass auf ihrem Server die JavaBeans mit Hilfe der lokalen Verzeichnisstruktur sinnvoll kategorisiert werden.)
- Name der JavaBean (Der <BeanName> unter dem eine JavaBean verzeichnet ist (<BeanName>.JAR-File). Sofern ein <BeanName> existiert, darf keine andere JavaBean mehr diesen Namen benutzen, es sei denn sie befindet sich in einer anderen Kategorie. Diese Einschränkung gilt allerdings nur für einen einzelnen Server.)

An der NewBeanBox selbst wird nur die ToolBox durch eine neue ToolBox mit der erweiterten Funktionalität ersetzt.

Die ToolBox wird initial mit den lokalen JavaBeans gefüllt. Sofern der Benutzer eine Verbindung zum Internet herstellt, kann er eine Verbindung zum Meta-Server oder einem der Bean-Server aufbauen. Beim Start der NewBeanBox überprüft die ToolBox allerdings nur, welche JavaBeans lokal vorhanden sind. Will der Benutzer die entfernten JavaBeans ebenfalls angezeigt bekommen, muss er online sein und den "Load Beans"-Button drücken.

3.3.1.1 Oberfläche

Die Oberfläche der ToolBox besteht aus zwei "Karteikarten":

- BEANS:
  Hier werden die gesamten JavaBeans in einer Baumstruktur dargestellt, und zwar geordnet nach verschiedenen Kategorien. Falls vorhanden, werden zu den JavaBeans die dazugehörigen Icons im "Bean-Baum" angezeigt. Andernfalls werden Standard-Symbole angezeigt.

  Diese Kategorien sind zum Beispiel:
  GUI/Frame (/Button/ Menu/ Dialog/ ...)
  GUI/Misc
  Animations
  Misc

  Im unteren Teil befinden sich zwei Buttons, einer zum Verbinden mit dem Meta-Server und einer zur Konfiguration der Serververbindung und einiger Systemeinstellungen.

- SEARCH:
  Hier stehen dem Benutzer diverse Suchmöglichkeiten zur Verfügung. Das Suchergebnis wird ebenfalls in einer geordneten Baumstruktur dargestellt.

3.3.1.2 BEANS-Kartei

Load Beans-Button Drückt der Benutzer den "Load Beans"-Button, während er online ist, wird versucht eine Verbindung zum Meta-Server aufzubauen. Die ToolBox wird zunächst mit den JavaBeans des Meta-Servers aktualisiert. Anschließend werden die vom Meta-Server gelieferten Adressen der Bean-Server kontaktiert. Diese liefern ihrerseits Informationen über ihre lokalen JavaBeans, die dann ebenfalls in die ToolBox eingetragen werden. Die Baumstruktur der ToolBox wird gemäß der Kategorien der aus dem Netz gelieferten Bean-Informationen auf den neusten Stand gebracht. Ist der Benutzer offline, so bekommt er eine Fehlermeldung und die ToolBox zeigt nur noch die lokalen JavaBeans an.

Configuration-Button Durch Drücken des "Configuration"-Buttons erscheint ein Dialogfenster, in dem die nötigen Einstellungen für die Verbindung zum Meta-Server eingestellt werden können. Das sind die Angabe der IP-Adresse eines Meta-Servers, dessen Port-Adresse und die Eingabe des Browsers mit dem die Javadoc-Files (HTML) angezeigt werden sollen. Das Eingabe-Feld für die IP-Adresse wird eine Combo-Box sein, in der direkt eine IP eingetragen wird oder durch ein Auswahlmenü ein Bean-Server ausgewählt werden kann. Diese Funktionalität ist für den Fall gedacht, dass der Meta-Server nicht erreichbar ist. Dann hat der Nutzer die Möglichkeit, einen Bean-Server aus der Liste auszuwählen, um überhaupt JavaBeans aus dem Netz zu

bekommen. Voraussetzung hierfür ist, dass vorher einmalig eine Verbindung zum Meta-Server aufgebaut wurde, damit der ToolBox die vorhandenen Bean-Server bekannt sind. (Die Adressen der Bean-Server werden lokal von der ToolBox gespeichert; initial kennt die ToolBox nur den Meta-Server). Der Name des Standardbrowser gibt an, welchen Browser der Benutzer installiert hat und welchen Browser die ToolBox verwenden soll, um die Info-Dateien der JavaBeans anzuzeigen. Als Voreinstellung wurde von der BeanDL-Gruppe Netscape gewählt.

In der Implementierungsphase hat sich gezeigt, das es sinnvoll ist noch einige für das eigene System wichtige Daten dort einzutragen. Zu diesen Daten gehören der Jars-Pfad und der Temp-Pfad. Der Jars-Pfad gibt an, in welchem Verzeichnis der Jars-Ordner liegt, der die lokalen JavaBeans enthält. Als Voreinstellung sucht sich die ToolBox den Pfad von dem aus die NewBeanBox gestartet wurde. Bei dem Temp-Pfad verhält es sich genauso, nur das die ToolBox dann weiß, wo sie temporäre Dateien anlegen soll.

3.3.1.3 Search-Kartei

Der Benutzer kann am unteren Rand der "Karteikarte" einen Suchbereich (Titel, Autor, Beschreibung (Volltext)) auswählen, seine Suche eingeben und diese dann mit dem "Find"-Button abschicken. Die Daten werden dann an den Meta-Server und die Bean-Server geschickt, die in ihren lokalen Bean-Verzeichnissen nach den passenden JavaBeans suchen und die Daten der entsprechenden JavaBeans zurückschicken. Zusätzlich wird noch im lokalen Jars-Verzeichnis nach weiteren JavaBeans gesucht, die die Suchanfrage erfüllen. Die gesuchten Informationen werden aus dem <BeanName>.HTML-File gewonnen, das mit Hilfe von Javadoc vom Anbieter erstellt worden ist. Das <BeanName>.HTML-File sollte in dem <BeanName>.Jars-File enthalten sein.

Die Ergebnisse werden dann in der "Suchansicht" ausgegeben.

3.3.1.4 PopUp-Menü

Über die rechte Maustaste steht dem Benutzer ein Popup-Menü mit den folgenden Einträgen zur Verfügung:

Info
: Durch Auswahl von INFO wird dem Benutzer die HTML-Dokumentation (das <BeanName>.HTML-File) zu der gerade angewählten JavaBean in einem Browser-Fenster angezeigt. Hierzu wird das entsprechende <BeanName>.HTML-File vom Meta-Server oder dem Bean-Server geholt, wenn es nicht auf dem lokalen Rechner liegt.

Download bzw. **Delete** - je nach Kontext
: Falls die gerade angewählte JavaBean eine lokale JavaBean ist, steht dem Benutzer die Möglichkeit zur Verfügung, diese mittels "DELETE" von seiner Festplatte zu löschen. Anschließend ist der Eintrag natürlich nicht mehr in der ToolBox enthalten.

  Im anderen Fall kann sich der Benutzer die angewählte JavaBean mittels "DOWNLOAD" vom Meta- oder Bean-Server auf die eigene Festplatte herunterladen. Die Applikation schickt hierzu einen HTTP-Request an den Meta-Server oder den Bean-Server und lädt das <BeanName>.JAR-File vom Server herunter. Das <BeanName>.HTML-File, das in dem <BeanName>.JAR-File enthalten ist, enthält die Informationen, die für die Suche

auf den lokalen JavaBeans verwendet werden. Das File wird dann in dem entsprechenden Jars-Verzeichnis abgelegt, und zwar unter der Kategorie, unter der es auch auf dem Server liegt.

3.3.1.5 Abschließende Anmerkungen zur NewBeanBox

Die ursprüngliche Funktionalität der BeanBox soll weitestgehend beibehalten werden, um dem BeanBox-erfahrenen Benutzer keine neue Arbeitsweise aufzuzwingen.

Das Auswählen einer JavaBean und die Benutzung verändert sich nicht. Der Benutzer wählt eine JavaBean aus der ToolBox mittels linker Maustaste aus und klickt anschließend in das BeanBox-Fenster, wo diese dann benutzt werden kann. Zusätzlich ist es innerhalb der Baumstruktur natürlich wie z.B. im Explorer möglich, einzelne Unterverzeichnisse auszublenden.

3.3.2 Server

Es werden Meta- und Bean-Server unterschieden. Von der Funktionalität unterscheiden sich Meta-Server und Bean-Server nicht. Als Meta-Server wird der Server bezeichnet, der zusätzlich eine Möglichkeit zum Hochladen von JavaBeans anbietet. Der Meta-Server stellt den KleinAnbietern von neuen JavaBeans ein BeanDL-Webinterface zur Verfügung.

Die Server (Meta-Server und Bean-Server) besitzen intern Listen über die ihnen bekannten Server. Wenn ein Bean-Server gestartet wird, meldet er sich bei dem ihm bekannten Meta-Server an. Bei der Anmeldung trägt der Meta-Server den neuen Bean-Server in seine Server-Liste ein. Nach festgelegten Zeitintervallen gleichen die Server (Meta- und Bean-Server) ihre Server-Listen ab. Wenn der Abgleich der Server-Listen vollständig ist, kennen sich alle Server untereinander.

Um einen Bean-Server zu installieren, muss man sich als erstes die entsprechende Software vom Meta-Server holen. Dieser stellt eine Homepage zur Verfügung, worüber dies einfach zu bewerkstelligen ist. Man lädt sich also von dort ein entsprechendes File herunter und installiert dieses auf seinem Server. Dann startet man den Bean-Server und dieser nimmt Kontakt mit dem Meta-Server auf um sich anzumelden. Nach der Anmeldung ist der eigene Bean-Server bereit, und es können JavaBeans auf ihm installiert werden. Diese kommen dann in das entsprechende Kategorie-Verzeichnis.

3.3.3 Das BeanDL - Webinterface

Das BeanDL - Webinterface wurde beim Entwurf zunächst noch nicht so konkret durchdacht, da der Schwerpunkt der BeanDL-Gruppe auf der Nutzungsseite (also beim Entwurf einer NewBeanBox + Server und deren erweiterter Funktionalität) lag und nicht auf der Anbieterseite. Trotzdem war klar, dass auf diesen Teil auch nicht verzichtet werden sollte, da es ja auch Klein-Anbieter gibt, die keinen eigenen BeanServer betreiben können, weil sie entweder nicht über die technischen Voraussetzungen wie einen eigenen WebServer oder entsprechenden Webspace mit der Möglichkeit, den BeanServer dort ausführen zu dürfen, verfügen. Oder sie haben einfach kein Interesse, einen eigenen BeanServer zu betreiben.

Mittels des BeanDL - Webinterfaces sollte auch der KleinAnbietergruppe die Möglichkeit gegeben werden, ihre eigenen selbsterstellten JavaBeans auf unserem MetaServer ablegen zu können.

Da uns anfangs noch nicht klar war, wie und ob man es realisieren kann, zusätzliche Dateien in die JavaBean mit hineinzupacken, sollte der Anbieter von eigenen JavaBeans zunächst folgende Dateien mit seiner JavaBean bereitstellen:

- seine JavaBean als JAR-File, z.B. Button.jar
- zusätzlich noch ein HTML-File, welches ein Info-File über die JavaBean sein sollte, und das bis auf die Datei-Endung genauso heissen sollte, wie das JAR-File, in diesem Fall also Button.html
- optional konnte noch eine Grafikdatei mit angegeben werden, die ebenfalls den gleichen Namen wie das JAR-File tragen sollte und auf .GIF enden sollte, hier also Button.gif

Zunächst war also die Registrierung folgendermassen geplant:

3.3.3.1 erster Entwurf: Registrierung

Beim ersten Entwurf wurde nur die Möglichkeit der Registrierung eigener JavaBeans bedacht und dementsprechend auch nur eine Schnittstelle definiert. Mittels eines Registrierungsformulars sollte der KleinAnbieter seine JavaBeans in unserer Bean Digital Library registrieren können. Hierzu sollte der KleinAnbieter einen Beannamen, die URL, wo die Dateien abgelegt wurden, seine E-Mail-Adresse, sowie ein frei ausgedachtes Passwort (doppelt) in das Formular eintragen und eine Kategorie auswählen, in die seine JavaBean passt.

Die Eingaben des KleinAnbieters sollten durch ein Servlet überprüft werden und die Daten von einem Web-Server, auf dem der KleinAnbieter seine JavaBean für kurze Zeit abgelegt hatte, heruntergeladen und auf unserem MetaServer abgelegt werden. Im Fehlerfall sollte eine Fehlermeldung ausgegeben werden und das Formular erneut vorgesetzt werden. Im Erfolgsfall sollte dem KleinAnbieter angezeigt werden, dass die Registrierung seiner JavaBean in unserer BeanDL erfolgreich war.

3.3.3.2 Änderungen und Erweiterungen

Nachdem uns die Realisierung einer neuen JavaBean inkl. Info-File und ggf. Grafik-File möglich war, haben wir noch mal die Definition unserer JavaBeans neu festgelegt. Nun sollte also die eigentlich JavaBean zusammen mit dem Info-File und ggf. dem Grafik-File zusammen in einem JAR-File abgelegt werden, was den Aufwand beim Übertragen bzw. Upload erheblich verringert hat. Aus dem JAR-File sollte sich dann auch sofort der JavaBeanname ergeben.

Im Zuge dieser Umstellung wurde natürlich auch das Webinterface und besonders die Registrierung neu überdacht und es ergaben sich die folgenden Änderungen und Erweiterungen, die im Einzelnen weiter unten noch genauer erläutert werden:

- bei der Registrierung muss der JavaBeanname nicht mehr explizit angegeben werden
- das Webinterface wurde um die folgenden Funktionalitäten erweitert:
  - Updaten von eigenen JavaBeans

- Löschen von eigenen JavaBeans
- Downloadmöglichkeiten für BeanServer- und NewBeanBox-Software

Diese Erweiterungen insbesondere des Updates und des Löschens waren sicherlich eher von Vorteil, denn sonst würden möglicherweise eine Vielzahl gleicher JavaBeans in den verschiedensten Versionen auf dem Meta-Server abgelegt werden, oder JavaBeans die versehentlich in die falsche Kategorie eingeordnet wurden in mehreren Kategorien abgelegt werden, was den freien Plattenspeicher auf die Dauer ziemlich dezimieren würde und somit später eventuell dazu führen könnte, dass neue JavaBeans nicht mehr registriert werden können, da kein Speicherplatz mehr zur Verfügung steht. Außerdem hätte ein KleinAnbieter keine Chance gehabt, seine eigene JavaBean der Öffentlichkeit wieder vorzuenthalten, wenn er sie einmal registriert hätte.

3.3.3.3 neue Registrierung

An der Registrierung wurde nicht viel zur alten Version verändert. Lediglich die Anpassungen an das neue Bean-Format (s.o) führte dazu, dass im Registrierungsformular die Eingabe des JavaBeannamens entfällt und bei der Angabe der URL die komplette URL inkl. des JAR-Files angegeben werden muss. Nun muss das Servlet nur noch das eine JAR-File vom Web-Server, wo der KleinAnbieter seine JavaBean für kurze Zeit für den Upload bereithalten sollte, herunterladen und die Einträge aus dem Registrierungsformular in die interne Datenbank übernehmen. Die Einträge werden beim Update bzw. beim Löschen wieder abgefragt, da die E-Mail-Adresse als Login und das Passwort zur Identifizierung benutzt werden. Doppelte JavaBeans in einer Kategorie können nicht entstehen, weil dieses während der Registrierung überprüft wird. Ansonsten hat sich an der Funktionalität der Registrierung nichts geändert.

3.3.3.4 Updaten einer JavaBean

Zusätzlich steht dem KleinAnbieter nun die Möglichkeit, seine eigenen JavaBeans über das WebInterface und ein entsprechendes Update-Formular upzudaten (also zum Beispiel eine neuere Version in unsere Bean Digital Library zu stellen und gegen die alte Version auszutauschen) oder einfach in eine neue Kategorie einzuordnen, zur Verfügung.

Hierfür muss der KleinAnbieter das Update-Formular ausfüllen, welches aus den folgenden Feldern besteht:

- alte URL der JavaBean
- neue URL der JavaBean (falls nicht alte URL)
- alte Kategorie
- neue Kategorie (oder “no change”, falls alte Kategorie)
- E-Mail-Adress (als Login)
- Passwort (aus der Registrierung)

Das Update funktioniert indem die alte JavaBean gelöscht wird (s. Löschen) und die neue JavaBean neu registriert wird (s. Registrieren). Hier wird ebenfalls im Fehlerfall eine Fehlermeldung ausgegeben und das Update-Formular erneut vorgesetzt. Im Erfolgsfall wird dem KleinAnbieter eine Rückmeldung gegeben.

3.3.3.5 Löschen einer JavaBean

Eine weitere neue Funktionalität ist das Löschen von JavaBeans aus unserer Bean Digital Library. Der KleinAnbieter hat die Möglichkeit seine eigenen JavaBeans über das Webinterface und das Löschen-Formular aus der BeanDL und vom MetaServer zu löschen und somit den anderen Nutzern wieder vorzuenthalten.

Hierfür muss er in dem Löschen-Formular die folgenden Felder ausfüllen:

- die URL seiner JavaBean
- die Kategorie, in der seine JavaBean abgelegt wurde
- seine E-Mail-Adresse (als Login)
- sein Passwort (aus der Registrierung)

Mittels der Daten aus dem Löschen-Formular wird überprüft, ob die entsprechende JavaBean in unserer BeanDL registriert ist und ob sie auf dem Meta-Server abgelegt ist. Auf dem Meta-Server wird das entsprechende File entfernt und der Datensatz aus der BeanDL gelöscht. Wie schon bei der Registrierung bekommt der KleinAnbieter im Fehlerfall eine Fehlermeldung und das Formular erneut vorgesetzt bwz. im Erfolgsfall eine Rückmeldung.

3.3.3.6 Downloadmöglichkeiten

Auf dem Webinterface soll auch noch die Möglichkeit bestehen, die Software für den BeanServer oder die NewBeanBox herunter zu laden. In diesen Software-Paketen, die in gepackter Form zur Verfügung gestellt werden sollen, ist dann auch eine entsprechende Anleitung zum Installieren und "In Betrieb nehmen" enthalten.

3.3.4 Use-Cases

3.3.4.1 Einleitung

An der eigentlichen BeanBox ändert sich vor allem die ToolBox. Ansonsten ändern sich nur Programmteile, die für den Nutzer nicht von Interesse sind, beziehungsweise bei der Ausführung des Programms nicht sichbar sind (z.B.: BeanLoader). Die Neuerungen befinden sich auf der Server-Seite, denn sowohl Meta-Server als auch Bean-Server existierten vorher nicht. Es gibt zwei Möglichkeiten, um neue JavaBeans zur Verfügung zu stellen. Entweder man besorgt sich den Bean-Server und stellt seine JavaBeans mit dessen Hilfe ins Netz, oder man meldet seine JavaBeans beim Meta-Server an und dieser holt sich diese aus dem Netz. Um die JavaBeans in Kategorien einzuteilen, wird pro Kategorie ein Unterverzeichnis im Verzeichnis "jars" angelegt. Dies gilt auf Seiten der ToolBox geanuso wie auf Seiten des Meta-Servers und der Bean-Server. Neue Kategorien kommen hinzu, wenn ein Anbieter auf Seiten des Bean-Servers in seinem "jars" Verzeichnis einen neuen Ordner anlegt.

3.3.4.2 Akteure

- Nutzer: Der Anwender der unsere BeanBox bei sich laufen hat und JavaBeans aus dem Internet (von Meta-Server oder einem der Bean-Server) benutzt.
- KleinAnbieter: Stellt JavaBeans zur Verfügung, hat aber keinen eigenen Web-Server laufen. Er gibt dem Meta-Server bekannt wo seine JavaBeans abgelegt sind. Von dort kann sich der Meta-Server die JavaBeans runterladen.
- Anbieter: Stellt seine JavaBeans der Öffentlichkeit mit Hilfe eines eigenen Bean-Servers zur Verfügung.
- ToolBox: Veränderte Version der ToolBox. Die ToolBox nimmt Verbindung mit den Servern auf, sofern der Nutzer online ist und auf "Load Beans" klickt. Über die ToolBox kann der Nutzer dann sehen welche Beans über das Netz geladen wurden und welche bei ihm lokal liegen. Diese JavaBeans kann der Nutzer dann in der BeanBox benutzen.
- BeanBox: Die BeanBox mit der bekannten Funktionalität
- Meta-Server: Auf dem Meta-Server tragen sich neue Bean-Server ein. Und KleinAnbieter können dort ihre JavaBeans unterbringen. Des weiteren stellt der Meta-Server über eine Homepage Informationen für den Anbieter und den KleinAnbieter zur Verfügung.
- Bean-Server: Der Bean-Server ist dem Meta-Server sehr ähnlich, bis auf die Bereitstellung einer Homepage und dadurch ergibt sich die Nichtannahme von JavaBeans der KleinAnbieter. Außerdem muss sich ein Bean-Server beim Meta-Server anmelden, um in die digitale Bibliothek aufgenommen zu werden.
- HTTP-Server des KleinAnbieters: HTTP-Server auf dem die JavaBeans der KleinAnbieter liegen, von wo der Meta-Server sich die JavaBeans runterlädt.

3.3.4.3 Auswahl einer JavaBean

Es wird beschrieben, wie eine JavaBean in der BeanBox vom Nutzer markiert wird. Akteure:
Nutzer
ToolBox

1. Der Nutzer wählt eine der zwei Karteikarten, "Beans" oder "Search", in der ToolBox aus.
2. Um eine JavaBean auszuwählen, klickt der Nutzer auf den Kategorieordner des Bean-Baums, in dem er die gewünschte JavaBean vermutet. Befindet sich der Nutzer in der Karteikarte "Search" muss er erst eine Suchanfrage eingeben und abschicken bevor er Beans auswählen kann. (Use-Case: Suchen einer JavaBean)
3. In den einzelnen Ordnern muss er sich unter Umständen weiter durchklicken.
4. Zum Schluss muss die eigentliche JavaBean durch anklicken mit der Maus markiert werden.
5. JavaBeans die lokal vorhanden sind, sind speziell markiert, während JavaBeans, die im Netz liegen, nicht weiter markiert sind.

3.3.4.4 Benutzung einer JavaBean

Es wird beschrieben, wie eine JavaBean in der BeanBox verwendet wird. Akteure:
Nutzer
ToolBox
BeanBox
Meta-Server
Bean-Server

1. Der Nutzer wählt in der ToolBox eine JavaBean aus. (Use-Case: Auswahl einer JavaBean)
2. Dann klickt er in die BeanBox, wo die JavaBean verwendet werden soll.
3. Die BeanBox holt sich die JavaBean entweder vom lokalen System oder vom Meta-Server oder einem der Bean-Server.
4. Dann kann der Nutzer mit der BeanBox und der JavaBean wie gewohnt arbeiten, das heißt wie er es mit der alten BeanBox des BDK 1.1 gewohnt ist.

3.3.4.5 Download einer JavaBean

Es wird beschrieben, wie eine JavaBean, die im Netz angeboten wird, auf dem lokalen System abgespeichert werden kann. Akteure:
Nutzer
ToolBox
Meta-Server
Bean-Server

1. Der Benutzer wählt in der ToolBox eine JavaBean aus. (Use-Case: Auswahl einer JavaBean)
2. Er wählt aus dem Pop-Up Menü die Option "Download" aus, was bei lokalen JavaBeans nicht möglich ist.
3. Die ToolBox lädt das <BeanName>.JAR-File vom Meta-Server oder von einem der Bean-Server herunter, und speichert das File im jars-Verzeichnis, das vom Benutzer angelegt wurde, ab. Dabei trägt sich das File automatisch in das richtige Kategorieverzeichnis ein.

3.3.4.6 Info über eine JavaBean

Es wird ein Fenster mit der HTML-Dokumentation zu einer JavaBean geöffnet.

Akteure:
Nutzer
ToolBox
Meta-Server
Bean-Server

1. Der Benutzer wählt in der ToolBox eine JavaBean aus. (Use-Case: Auswahl einer JavaBean)
2. Er wählt aus dem Pop-Up Menü die Option "Info" aus.
3. Das <BeanName>.HTML-File wird vom Meta-Server oder einem der Bean-Server geladen, sollte die JavaBean auf dem lokalen System liegen, wird natürlich dieses <BeanName>.HTML-File benutzt.
4. Dieses Dokument wird in einem Browser-Fenster angezeigt.

3.3.4.7 Löschen einer JavaBean

Es wird eine JavaBean im lokalen System gelöscht.

Akteure:
Nutzer
ToolBox

1. Der Benutzer wählt in der ToolBox eine als lokal markierte JavaBean aus. (Use-Case: Auswahl einer JavaBean)
2. Er wählt aus dem Pop-Up Menü die Option "Löschen" aus.
3. Die JavaBean (<BeanName>.JAR-File) wird aus dem lokalen System entfernt.
4. In der ToolBox wird die JavaBean entfernt, wenn sie nicht im Netz vorhanden ist. Ansonsten wird der Eintrag lokal entfernt.

3.3.4.8 Laden einer JavaBean

Die ToolBox fragt beim Meta-Server und den Bean-Servern die vorhandenen JavaBeans ab.

Akteure:
Nutzer
ToolBox
Meta-Server
Bean-Server

1. Der Nutzer klickt in der ToolBox, in der Karteikarte "Beans" auf "Load Beans".
2. Die ToolBox nimmt Kontakt mit dem Meta-Server auf.
3. Der Meta-Server schickt die Daten der vom Meta-Server erhältlichen JavaBeans und die Adressen der angemeldeten Bean-Server zurück.
4. Die ToolBox nimmt Kontakt mit den Bean-Servern auf.
5. Die ToolBox erhält die Daten der auf den Bean-Servern erhältlichen JavaBeans zurück.
6. Der BeanBaum in der ToolBox wird nach jeder Rückmeldung der verschiedenen Server aktualisiert.

3.3.4.9 Suchen einer JavaBean

Die ToolBox fragt im lokalen System, beim Meta-Server und bei den Bean-Servern die zu den Suchkriterien passenden Daten ab.

Akteure:
Nutzer
ToolBox
Meta-Server
Bean-Server

1. Der Nutzer gibt in der ToolBox, in der Karteikarte "Search" sein Suchkriterium ein.
2. Der Nutzer klickt in der ToolBox, in der Karteikarte "Search" auf "Search".
3. Die ToolBox nimmt mit dem Meta-Server Kontakt auf und schickt die Suchanfrage an den Meta-Server.
4. Die ToolBox nimmt mit den Bean-Servern Kontakt auf und schickt die Suchanfrage an die verschiedenen Bean-Server.
5. Der Meta-Server und die Bean-Server suchen auf ihren JavaBeans nach den passenden Daten.
6. Alle erfolgreichen Suchergebnisse werden an die Toolbox zurückgeschickt.
7. Die ToolBox aktualisiert den BeanBaum in der Karteikarte "Search", in diesem Fall ebenfalls nach jeder Rückmeldung der einzelnen Server.

3.3.4.10 Konfiguration der ToolBox

Der Nutzer kann die Grundeinstellungen der NewBeanBox verändern dazu gehören die Servereinstellungen, das sind die Daten für den Meta-Server, die Browserinformationen, welchen Browser verwendet das eigene Szstem, den Jars-Pfad, in welchem Verzeichnis liegen die lokalen JavaBeans, und den Temp-Pfad, hier kann der Nutzer einstellen.

Akteure:
Nutzer
ToolBox

1. Der Nutzer klickt in der ToolBox, in der Karteikarte "Beans" auf "Configuration".
2. Es erscheint ein Fenster in dem der Nutzer die IP-Adresse und die Port-Nummer des Meta-Servers festlegt. Sollte dieser mal nicht antworten kann er auch hier die Daten eines Bean-Servers eintragen oder auswählen. Diese Daten über andere Bean-Server bekommt er dann zur Auswahl, wenn er schon einmal die NewBeanBox gestartet hat und mit dem Meta-Server vebunden war. Außerdem kann der Nutzer den verwendeten Browser, den Jars-Pfad und den Temp-Pfad eingeben. In dem Feld Browser gibt er an, mit welchem Browser er arbeitet, beziehungsweise welcher Browser auf seinem Rechner installiert ist. Voreingestellt ist hier Netscape. Der Jars-Pfad ist die Angabe des Pfades, der zu seinem Jars-Ordner

führt, in dem seine Beans liegen. Voreingestellt ist hier der Pfad von dem aus der Nutzer die NewBeanBox startet. Der Temp-Pfad ist die Angabe des Pfades, der zu seinem Temp-Ordner führt, in dem die temporären Dateien des Nutzers liegen. Voreingestellt ist hier der Pfad von dem aus der Nutzer die NewBeanBox startet.

3.3.4.11 Anmeldung einer JavaBean durch den KleinAnbieter beim Meta-Server

Es wird eine JavaBean durch den KleinAnbieter auf dem Meta-Server angemeldet.

Akteure:
KleinAnbieter
Meta-Server
HTTP-Server des KleinAnbieters

1. Der Anbieter wählt auf der Homepage des Meta-Servers den Link “Registration Page“.
2. Auf der nun folgenden Seite trägt der Anbieter die URL der JavaBean ein, er gibt den Namen der JavaBean an, wählt eine Kategorie aus, gibt seine E-Mail-Adresse und ein Passwort (zur Sicherheit zweimal) ein.
3. Mit Klick auf den Button “Register Bean“ wird die JavaBean auf dem Meta-Server angemeldet.
4. Der Meta-Server überprüft, ob an der angegebenen URL ein entsprechendes File vorhanden ist.
5. Der Meta-Server lädt sich das <BeanName>.JAR-File herunter.
6. Der Meta-Server lädt das File in ein “jars“-Verzeichnis das eine Struktur hat, die alle existierenden Kategorien enthält. Dort trägt der Meta-Server das File in das entsprechende Kategorie-Verzeichnis ein.

3.3.4.12 Updaten einer JavaBean durch den KleinAnbieter beim Meta-Server

Der KleinAnbieter einer JavaBean kann diese jederzeit updaten.

Akteure:
KleinAnbieter
Meta-Server
HTTP-Server des KleinAnbieters

1. Der KleinAnbieter wählt auf der Homepage des Meta-Servers den Link “Update page“.
2. Auf der nun folgenden Seite trägt der KleinAnbieter den Namen der JavaBean ein, unter der die JavaBean auf dem Meta-Server gespeichert ist. Natürlich kann er nur JavaBeans updaten, die er selbst ins Netz gestellt hat.
3. Dann hat er die Möglichkeit eine URL einzugeben, unter der die Update Version zu finden ist.

4. Der KleinAnbieter wählt eventuell eine neue Kategorie aus, .
5. Dann gibt er seine E-Mail-Adresse und sein Passwort zur Authentifizierung ein.
6. Dann kann er mit Hilfe des Buttons "Update Bean" die Informationen an den Meta-Server abschicken.
7. Der Meta-Server entfernt das <BeanName>.JAR-File, aus dem entsprechenden KategorieVerzeichnis, und lädt die neuen Daten vom HTTP-Server des KleinAnbieters.

3.3.4.13 Löschen einer JavaBean durch den KleinAnbieter beim Meta-Server

Der KleinAnbieter einer JavaBean kann diese jederzeit löschen.

Akteure:
KleinAnbieter
Meta-Server

1. Der KleinAnbieter wählt auf der Homepage des Meta-Servers den Link "Remove page".
2. Auf der nun folgenden Seite trägt der KleinAnbieter den Namen der JavaBean ein, die vom Meta-Server gelöscht werden soll. Natürlich kann er nur JavaBeans entfernen, die er selbst ins Netz gestellt hat.
3. Dann gibt er seine E-Mail-Adresse und sein Passwort zur Authentifizierung ein.
4. Dann klickt er auf den Button "Remove Bean".
5. Der Meta-Server entfernt das <BeanName>.JAR-File, aus dem entsprechenden KategorieVerzeichnis.

3.3.4.14 Bean-Server installieren und anmelden (Anbieter)

Der Anbieter muss den Bean-Server bei sich installieren und der Bean-Server meldet sich beim Start beim Meta-Server an.

Akteure:
Anbieter
Meta-Server
Bean-Server

1. Der Anbieter lädt von der Meta-Server Homepage das Bean-Server Paket herunter.
2. Er installiert das Paket.
3. Er startet den Bean-Server auf seinem Online-Server.
4. Der Bean-Server meldet sich beim Meta-Server an.
5. Der Meta-Server trägt die Daten über den Bean-Server bei sich ein.

3.3.4.15 JavaBeans auf dem Bean-Server eintragen oder updaten (Anbieter)

Der Anbieter legt seine JavaBeans auf den Bean-Server.

Akteure:
Anbieter
Bean-Server

1. Der Anbieter stellt das <BeanName>.JAR-File, in das entsprechende KategorieVerzeichnis im Verzeichnis “jars“.
2. Sollte sich die Kategorie einer JavaBean verändern, so muss der Anbieter natürlich das alte File der JavaBean aus dem entsprechenden KategorieVerzeichnis löschen und das neue File in das entsprechend neue KategorieVerzeichnis einfügen.

3.3.4.16 JavaBeans vom Bean-Server löschen (Anbieter)

Der Anbieter einer JavaBean kann diese jederzeit löschen.

Akteure:
Anbieter
Bean-Server

1. Der Anbieter entfernt das <BeanName>.JAR-File, aus seinem KategorieVerzeichnis im Verzeichnis “jars“.

3.4 Designentscheidungen und Art der Implementierung

Zur Realisierung der erweiterten BeanBox wurde Java 1.2 von Sun Microsystems gewählt. Dabei wurde das Java Development Kit (JDK) 1.2.2 und die Bean Development Kit (BDK) 1.1 verwendet. Die Entscheidung fiel auf Java 1.2, da es, im Gegensatz zu den vorherigen Java-Versionen, die Implementierung der grafischen Benutzungsschnittstelle wesentlich vereinfacht und schneller zu annehmbaren Ergebnissen führt. Insbesondere die Implementierung von Baum-Komponenten wird mit Hilfe des Swing-Packets erheblich erleichtert, da eine entsprechende Komponente (JTree) in dem Packet enthalten ist. Dieser Aspekt kommt bei der Darstellung in der neuen ToolBox zum Tragen, wo die JavaBeans über ein Baumstruktur dargestellt werden.

Als Netzwerktechnologie ist Java-RMI zum Einsatz gekommen. Sämtliche Komponenten der erweiterten BeanBox sind in Java geschrieben, daher bot sich RMI als reine Java-Lösung an. Ein weiterer Grund für die Wahl von RMI ist, dass die Entwicklung von Client/Server-Applikationen hierüber relativ einfach zu realisieren ist.

Für die Gestaltung des BeanDL - Webinterfaces wurden HTML, JavaScript und Java für die Servlets benutzt. Die Funktionalitäten für die Registrierung, das Update und das Löschen wurden mittels Java-Servlets realisiert Man hätte sicherlich auch eine andere Technologie für das Ausprogrammieren der Funktionalitäten des BeanDL-Webinterfaces benutzen können, wie zum Beispiel CGI, allerdings wollten wir auch bei diesem Teil, wie schon bei der NewBeanBox, den Servern und der Netzwerktechnologie auf Java nicht verzichten. Somit boten sich Java-Servlets hier besonders an. Ein weiterer Aspekt war, daß die Funktionalität auf Seiten des Web-Servers gehalten werden sollte und somit auch Nutzern von Browser der älteren Generation (ohne clientseitige Java-Unterstützung) das Anmelden ihrer JavaBeans über das BeanDL-Webinterface möglich gemacht werden sollte.

Sun hat mit den Servlets - in Anlehnung an Applets auf Clientseite - ein Konzept der mobilen Java-Bytecode-Objekte für Server entwickelt, das die Vorzüge von Applets und CGI auf der Grundlage eines objektorientierten Ansatzes vereint. Das Konzept beschreibt eine einfache, robuste, plattformunabhängige Schnittstelle auf Serverseite, die eine Erweiterung der Funktionalität unabhängig vom Server und von der Plattform mit Java-Bytecode erlaubt.

Servlets sind zu einem spezifischen Interface konforme Java-Objekte. Sie ähneln Applets insofern, als sie - dynamisch über das Netz zu ladende - Bytecode-Objekte sind. Andererseits haben Servlets kein "Gesicht", das heißt, sie haben kein eigenes grafisches Interface.

Man unterscheidet lokale und entfernte Servlets. Entfernte Servlets sind wie Applets durch eine URL-Adresse identifizierbar, lokale durch ihren Klassennamen. Servlets lassen sich direkt (durch Angabe eines Servlet-Namen) oder indirekt (durch Verknüpfung mit Dokumenten) aufrufen. Ob Servlets beim Start des Servers starten oder dynamisch zur Laufzeit, entscheidet der Administrator unter "Servlet Loading".

Nachdem wir uns ein wenig mit der Servlet-Programmierung vertraut gemacht hatten und die Formulare auch soweit fertig waren, begann der schwierigere Teil - das Ausprogrammieren der Funktionalitäten. Hierbei wurden wir vor das Problem gestellt, dass wir nicht genau wussten, wohin die Fehlermeldungen geschrieben wurden und uns immer wieder mit Fehlermeldungen wie "Internal Server ..." auseinander setzen mussten. Erst nach einiger Zeit und etwas herumfragen und suchen haben wir dann herausgefunden, worauf die entstandenen Fehler zurückzuführen waren. Mit den entsprechenden Fehlermeldungen war die Fehlerkorrektur dann um einiges leichter durchzuführen.

3.5 Benutzerhandbuch

3.5.1 NewBeanBox

3.5.1.1 Installation

Die Installation der NewBeanBox beschränkt sich auf das Entpacken des Archivs, in welchem die NewBeanBox vertrieben wird. Dieses ist ein tar-Archiv welches mit dem Shell-Kommando "tar xvzf NewBeanBox.tgz" entpackt wird. Nach dem Entpacken liegen drei neue Dateien im Arbeitsverzeichnis:

NewBeanBox.jar Das Java-Archiv in dem sich alle Klassendateien der NewBeanBox befinden.

policy.txt Die Policy-Datei wird für den RMISecurityManager benötigt.

newbeanbox.sh Das Shell-Skript mit dem die NewBeanBox gestartet wird.

Für die Benutzung der NewBeanBox muss Java mindestens in der Version 1.2 auf Ihrem Rechner installiert sein. Falls die Version 1.2 von Java auf Ihrem Rechner durch ein anderes Kommando als einfach "java" gestartet wird, muss das Startskript entsprechend angepasst werden.

3.5.1.2 Benutzung

Programmstart
Der Start der NewBeanBox geschied durch das ausführen des Shell-Skriptes NewBeanBox.sh in der Kommandozeile.

Die BeanBox
An der BeanBox wurden mit Ausnahme der ToolBox keine Veränderungen vorgenommen die eine veränderte Benutzung erfordern, d.h. dass die Benutzung identisch ist zu der BeanBox des BDK 1.1 . Daher wird hier nur auf die Benutzung der neuen ToolBox eingegangen.

Die neue ToolBox

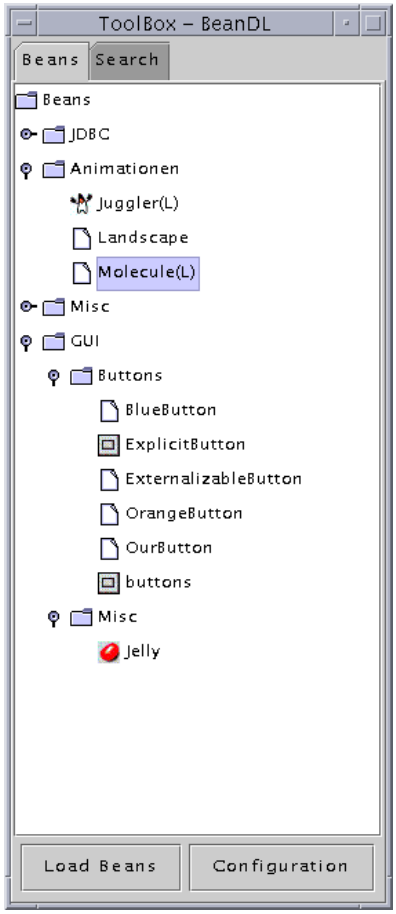

Abbildung 3.1: Die neue ToolBox

In der ToolBox (Abb. 3.1) werden die JavaBeans ausgewählt, die in der BeanBox benutzt werden sollen. Die Oberfläche der ToolBox besteht aus zwei "Karteikarten" , "Beans" und "Search". In beiden Ansichten werden die vorhandenen JavaBeans in einer Baumstrucktur dargestellt. Die lokal vorhandenen JavaBeans sind durch ein "(L)" hinter dem Beannamen gekennzeichnet.

Im Folgenden werden zunächst die Aktionen beschrieben, die sowohl in der "Beans" wie auch in der "Search-Karteikarte" möglich sind.

Auswahl und Benutzung von JavaBeans: Die Auswahl einer JavaBean geschieht durch Anklicken mit der Maus. Wenn eine JavaBean ausgewählt wurde, wird der Cursor zu einem Kreuz, wird jetzt in das Fenster der BeanBox geklickt, dann wird die ausgewählte JavaBean dort eingefügt.

Ein Klick mit der rechten Maustaste auf eine lokale JavaBean öffnet ein Popup-Menü (Abb. 3.2) mit den Einträgen:

Info: Die Auswahl von Info startet den in der Konfiguration eingestellten Browser mit der Beschreibung der JavaBean.

Delete: Bei den lokalen JavaBeans besteht durch diesen Menüpunkt die Möglichkeit, die JavaBean wieder zu löschen.

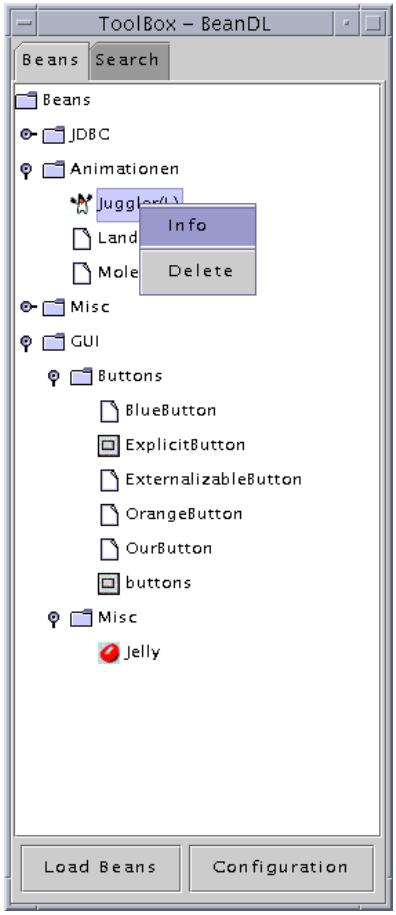

Abbildung 3.2: Das PopUp Menü (lokale JavaBean)

Ein Klick mit der rechten Maustaste auf eine nicht lokale JavaBean öffnet ein Popup-Menü (Abb. 3.3) mit den Einträgen:

Info: Die Auswahl von Info startet den in der Konfiguration eingestellten Browser mit der Beschreibung der JavaBean.

Download: Bei nicht lokalen JavaBeans besteht durch diesen Menüpunkt die Möglichkeit, eine Kopie der JavaBean auf den lokalen Rechner zu laden.

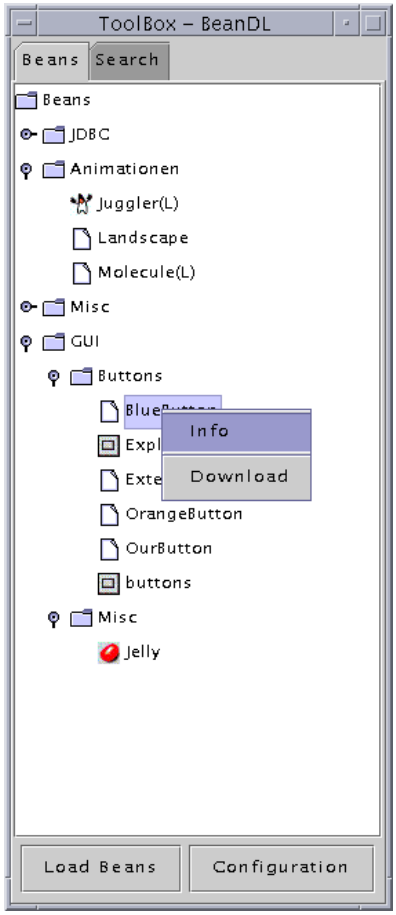

Abbildung 3.3: Das PopUp Menü (remote JavaBean)

Im unteren Teil der “Karteikarte“ “Beans“ befinden sich die Schaltflächen:

Load Beans: Durch betätigen dieses Schalters wird eine Verbindung zu den Servern aufgebaut und es werden die dort vorhandenen JavaBeans in die Baumstrucktur eingetragen.

Configuration: Durch betätigen dieses Schalters wird das Konfigurationsfenster (Abb. 3.4) geöffnet. Hier besteht die Möglichkeit verschiedene Einstellungen vorzunehmen:

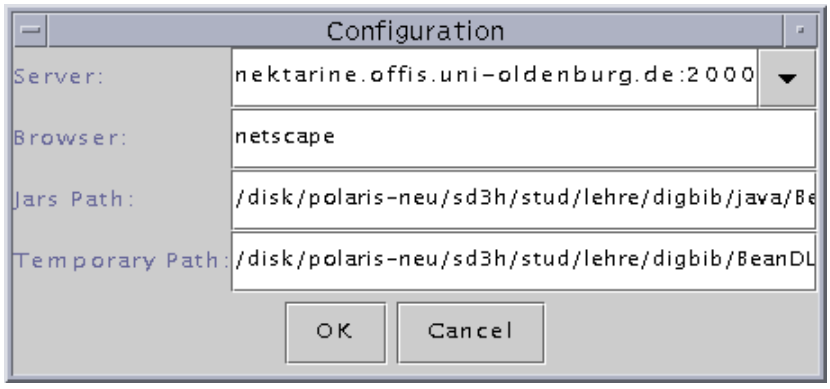

Abbildung 3.4: Die Konfigurationseinstellungen

Server: Hier kann der Server eingetragen werden, der als Metaserver benutzt werden soll. D.h. von diesem Server werden die Adressen der anderen Server angefordert.

Browser: Der hier eingetragene Browser wird für die Anzeige der Beanbeschreibung verwendet.

Jars Path: Hier werden auf dem lokalen Rechner die Jar Dateien gesucht und abgelegt.

Temporary Path: Hier werden die temporär benötigten Dateien angelegt.

Im unteren Teil der “Karteikarte“ “Search“ befinden sich die Felder (Abb. 3.5) für die Spezifikation der Suche, und ein Schalter um die Suchanfrage abzuschicken.

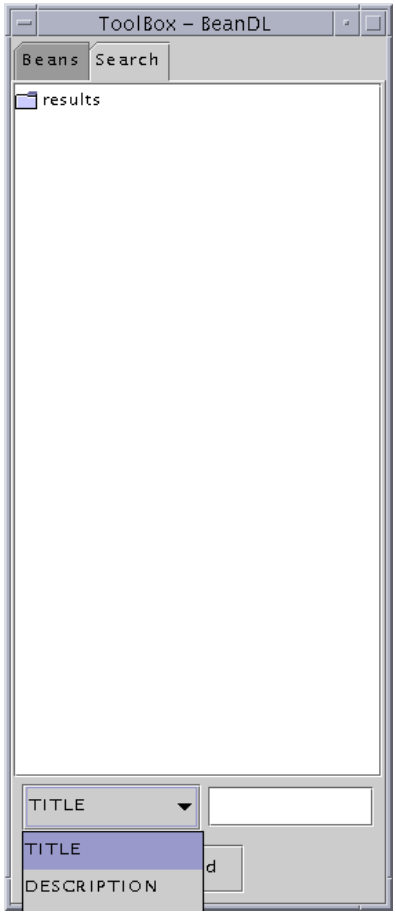

Abbildung 3.5: Die Suchmaske

Die Spezifikation der Suche: Hier kann die Suchanfrage spezifiziert werden: In der Auswahlbox wird der Bereich für die Suche eingestellt und in dem Textfeld wird eingegeben wonach gesucht werden soll. Der Bereich für die Suche umfasst den Namen des Autors, den Titel der JavaBean und die Beschreibung einer JavaBean, was einer Volltextsuche entspricht.

Suchanfrage starten: Durch betätigen dieses Schalters wird die Suchanfrage an die Server und an die ToolBox abgeschickt. Die Ergebnisse der Suche werden sobald sie eintreffen in die Baumstrucktur auf der “Search“ “Karteikarte“ eingefügt (Abb. 3.6).

Abbildung 3.6: Die Suchergebnisse

3.5.2 Installation und Inbetriebnahme des BeanServers

3.5.2.1 Vorbemerkungen

Zum Betrieb des BeanServers ist ein Betriebssystem notwendig, dass auf dem Host des Servers eine Runtime-Umgebung für die Java-Version 1.2 (z.B. JRE 1.2.2 von Sun) vorhanden ist. Eine weitere Voraussetzung ist, dass das TCP/IP-Protokoll vom Betriebssystem zur Verfügung gestellt wird. Für einen sinnvollen Einsatz des BeanServers empfiehlt sich natürlich eine Netzwerkanbindung des Hosts, sei es nun ein lokales Netzwerk oder das Internet.

Um den BeanServer im Netzwerk zugreifbar zu machen, ist es erforderlich, dass das Server-Packet (server.jar) auf eine bestimmte Art vom Host des Servers den Clients zugänglich gemacht wird. Dies wird über die Angabe einer URL beim Starten des Servers geregelt. In lokalen Netzwerken lässt sich dies über die Netzwerkfreigabe von Verzeichnissen regeln, im Internet empfiehlt sich hierfür ein HTTP- oder FTP-Server.

3.5.2.2 Installation

Die relevanten Dateien des Server sollten sich sinnvollerweise in einem gesonderten Verzeichnis befinden. Zum erfolgreichen Betrieb benötigt der BeanServer ein Unterverzeichnis für temporäre Dateien („tmp“) und ein Verzeichnis zum Abspeichern der JavaBeans (*.jar-Dateien). Das temporäre Verzeichnis muss in dem Start-Verzeichnis des Servers liegen (d.h., das Verzeichnis, aus dem der Server gestartet wird, muss dieses Unterverzeichnis enthalten) und muss „tmp“ lauten. Der Name des Jars-Verz. ist frei wählbar, da er beim Starten des Servers angegeben wird.

Das Server-Packet (server.jar) muss in ein von außerhalb zugängliches Verzeichnis kopiert werden. Dies kann, wie schon in den Vorbemerkungen erwähnt, ein Verzeichnis sein, wo HTTP- oder FTP-Server den Zugriff von außen erlauben.

3.5.2.3 Inbetriebnahme

Das Starten des BeanServers erfolgt in 2 Schritten:

1. Starten der RMI-Registry
   Der BeanServer nutzt die Java-RMI-Technologie. Ist daher erforderlich, vor dem Aufruf des Servers die RMI-Registry zu starten, damit sich der Server bei Registry anmelden kann. Folgend die Aufrufe für die Betriebssysteme Windows und Unix.

   Windows:

   ```
   start rmiregistry [Portnummer]
   ```

   Unix:

   ```
   rmiregistry [Portnummer]
   ```

   Die Angabe der Portnummer ist optional. Wird keine Portnummer angegeben, wird die RMI-Standardportnummer 1099 angenommen.

2. Starten des BeanServers
   Der Aufruf des BeanServers ist relativ aufwendig. Es ist daher ratsam, den Befehl über ein Shell-Skript auszuführen. Eventuelle Fehleingaben lassen sich so leichter korrigieren.

   Die Syntax des Serversaufrufs sieht wie folgt aus:

   ```
   java -cp <Pfad zu server.jar>
       -Djava.rmi.server.codebase=<URL server.jar>
       -Djava.security.policy=<Policy-Datei>
       Server.BeanServer <Portnummer>
       <Pfad zum Jars-Verz.> [<Ip-MetaServer:Portnummer>]
   ```

   Erläuterung der Server-Optionen:

   - `<Pfad zu server.jar>`
     Hier befinden sich die relavanten Klassen des Servers.
   - `<URL server.jar>`
     Um das dynamische Nachladen von Code zu ermöglichen, muss die Datei „server.jar“ von anderen Rechnern zu erreichen sein. Die entsprechende URL ist hier einzutragen.
   - `<Policy-Datei>`
     Hier ist der Pfad zur Policy-Datei einzutragen. Die Policy-Datei legt fest, welche Rechte einem Client gewährt werden. (z.B. auf welchem Port connected werden darf, bzw. Verbindungen akzeptiert werden)
   - `Server.BeanServer <Portnummer>`
     Die Portnummer wird benötigt, damit der Server sich bei der RMI-Registry eintragen kann. I.d.R. ist die Portnummer 1099.
   - `<Pfad zum Jars-Verz.>`
     Gibt an, in welchem Verzeichnis sich die JavaBeans (Jars) des Servers befinden.
   - `<Ip-MetaServer:Portnummer>`
     Wenn der Server sich bei einem MetaServer eintragen soll, muss an dieser Stelle die URL bzw. die IP-Nummer des MetaServers zusammen mit der entsprechenden Portnummer angegeben werden. Diese Angabe kann weggelassen werden, wenn der Server selbst MetaServer sein soll oder wenn keine anderen BeanServer bekannt sind.
     Wurde ein MetaServer angegeben, versucht der BeanServer, diesen zu kontaktieren und sich bei ihm einzutragen. Danach tauschen die Server im Minutentakt ihre Serverlisten aus und gewährleisten somit, dass sich alle Server untereinander kennen.

3.5.3 Das BeanDL - Webinterface

3.5.3.1 Download-Möglichkeiten

Auf dem BeanDL - Webinterface gibt es einen Link auf eine WWW-Seite, auf der die Möglichkeit besteht, sich die Software für die NewBeanBox und den eigenen Bean-Server herunter zu laden. So soll gewährleistet werden, dass jeder Anbieter von neuen JavaBeans, dem die technischen Voraussetzungen (eigener Server oder entsprechender Webspace mit der Möglichkeit darauf

einen Bean-Server laufen lassen zu können) zur Verfügung stehen, seinen eigenen Bean-Server installieren kann und dadurch seine eigenen Beans den anderen Nutzern der neuen BeanBox zur Verfügung stellen kann.

Ebenso können sich interessierte Nutzer die NewBeanBox-Software herunterladen und somit die neuen erweiterten Fähigkeiten bei sich zuhause nutzen.

3.5.3.2 Registrieren einer JavaBean

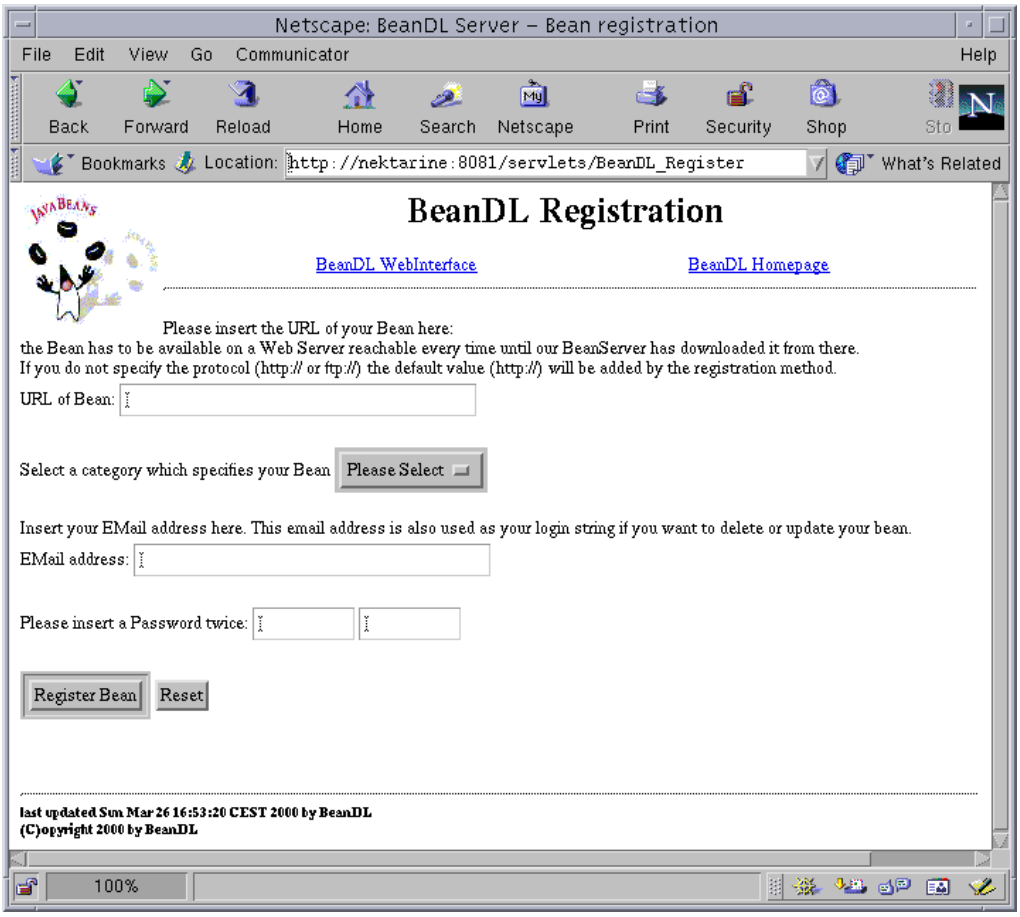
Netscape: BeanDL Server – Bean registration
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Sto
Bookmarks Location: http://nektarine:8081/servlets/BeanDL_Register What's Related
BeanDL Registration
BeanDL WebInterface
BeanDL Homepage
Please insert the URL of your Bean here:
the Bean has to be available on a Web Server reachable every time until our BeanServer has downloaded it from there.
If you do not specify the protocol (http:// or ftp://) the default value (http://) will be added by the registration method.
URL of Bean:
Select a category which specifies your Bean
Please Select
Insert your EMail address here. This email address is also used as your login string if you want to delete or update your bean.
EMail address:
Please insert a Password twice:
Register Bean
Reset
last updated Sun Mar 26 16:53:20 CEST 2000 by BeanDL
(C)opyright 2000 by BeanDL
100%

Abbildung 3.7: Die Registrierung einer JavaBean

Für das Registrieren einer neuen JavaBean steht dem KleinAnbieter eine eigene HTML-Seite zur Verfügung, in der er ein Formular (Abb. 3.7) ausfüllen muss, um seine neue JavaBean registrieren zu lassen und einen Upload auf den Meta-Server zu aktivieren. Dazu füllt der KleinAnbieter in dem Registrierungsformular die folgenden Felder aus:

- die URL, wo die neue JavaBean zum Upload für eine gewisse Zeit abgelegt wurde. Die URL kann sowohl mit http:// als auch mit ftp:// beginnen. Falls kein Protokoll angegeben wurde, so wird automatisch http:// hinzugefügt.

- die Kategorie, unter der die JavaBean auf dem Meta-Server abgelegt werden soll.
- die e-mail-Adresse des KleinAnbieters, die ebenfalls als Login für das Update oder Löschen einer JavaBean benutzt wird.
- ein Passwort, welches sich der KleinAnbieter selbst ausdenken darf und welches er zur Sicherheit doppelt eingeben muss.

Nachdem der KleinAnbieter seine Daten in die entsprechenden Felder des Registrierungsformulars eingegeben hat, hat er entweder durch drücken des RESET-Knopfes die Möglichkeit die Felder wieder auf den Anfangswert zurück zu setzen oder er schickt seinen Registrierungsauftrag durch drücken des REGISTER-Knopfes ab. An dieser Stelle beginnt die eigentliche Arbeit des Registrierungs-Servlets. Es liest die Werte der Formularfelder aus und überprüft ihren Inhalt auf Richtigkeit bzw. testet, ob die Felder alle ausgefüllt wurden und die beiden Passwörter übereinstimmen. Ist dies der Fall, so werden die Daten in der internen Datenbank (hier eine Datei) registriert und versucht die Bean von der angegebenen URL herunter zu laden.

Tritt irgendwo ein Fehler auf, so wird dem KleinAnbieter das Registrierungsformular erneut angezeigt und zusätzlich oben noch eine Fehlermeldung, die die Fehlerursache erläutert, mit ausgegeben. Falls die Registrierung und der Download erfolgreich waren, so bekommt der KleinAnbieter eine neue HTML-Seite vorgesetzt, auf der er über das erfolgreiche Registrieren seiner JavaBean in der BeanDL in Kenntnis gesetzt wird und er nun verschiedene Möglichkeiten hat, weiter zu surfen.

3.5.3.3 Update einer JavaBean

Wenn der KleinAnbieter seine eigene JavaBean updaten will, wählt auf dem BeanDL-WebInterface den Link "Update page". Er bekommt dann eine neue HTML-Seite mit einem Update-Formular (Abb. 3.8) zu sehen, welches er ausfüllen muss, um seine JavaBean upzudaten.

Ein Update kann sowohl eine Verlagerung einer JavaBean in eine andere Kategorie sein, oder aber auch, wenn eine neuere Version einer schon vorhandenen JavaBean vorliegt und diese den Nutzern zugänglich gemacht werden soll.

Das Update-Formular, welches der KleinAnbieter aus zu füllen hat, besteht aus den folgenden Feldern:

- die URL, wo die alte JavaBean zum Upload abgelegt war. Die URL kann sowohl mit http:// als auch mit ftp:// beginnen. Falls kein Protokoll angegeben wurde, so wird automatisch http:// hinzugefügt.
- optional die URL, wo die neue JavaBean zum Upload für eine gewisse Zeit abgelegt wurde. Hier brauch nichts eingetragen werden, wenn die JavaBean wieder unter der alten URL zu finden ist. Die URL kann sowohl mit http:// als auch mit ftp:// beginnen. Falls kein Protokoll angegeben wurde, so wird automatisch http:// hinzugefügt.
- die Kategorie, unter der die JavaBean auf dem Meta-Server abgelegt wurde.

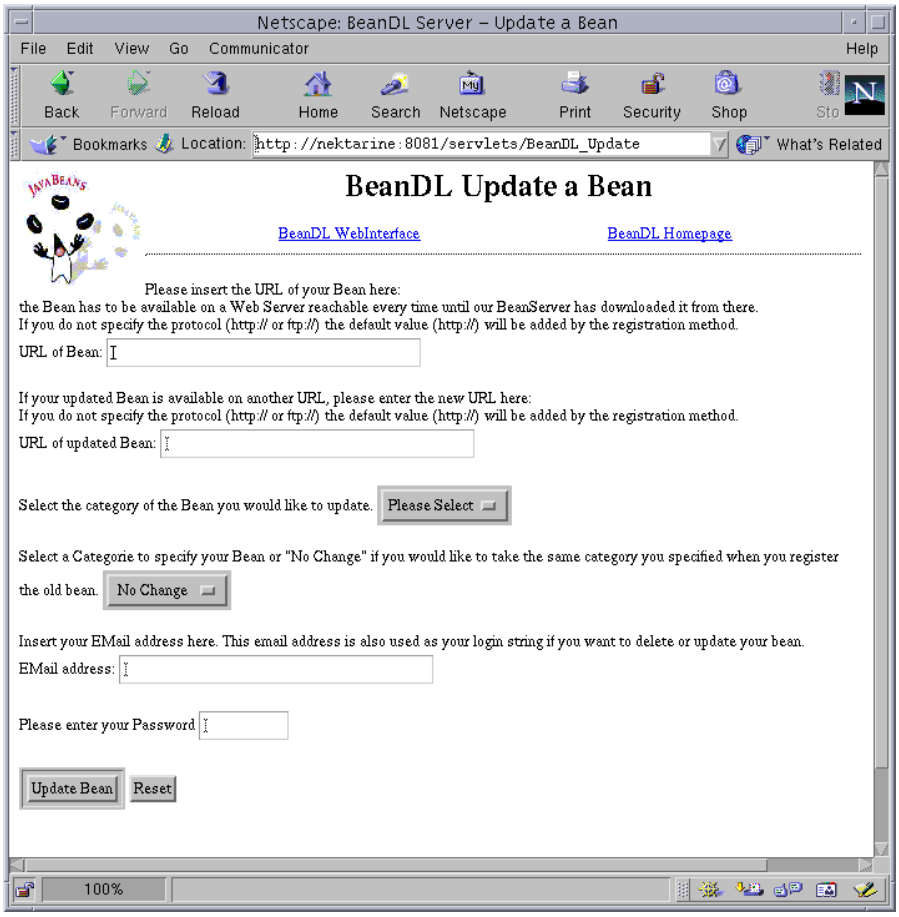
Netscape: BeanDL Server – Update a Bean
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Sto
Bookmarks Location: http://nektarine:8081/servlets/BeanDL_Update What's Related
BeanDL Update a Bean
BeanDL WebInterface
BeanDL Homepage
Please insert the URL of your Bean here:
the Bean has to be available on a Web Server reachable every time until our BeanServer has downloaded it from there.
If you do not specify the protocol (http:// or ftp://) the default value (http://) will be added by the registration method.
URL of Bean:
If your updated Bean is available on another URL, please enter the new URL here:
If you do not specify the protocol (http:// or ftp://) the default value (http://) will be added by the registration method.
URL of updated Bean:
Select the category of the Bean you would like to update.
Please Select
Select a Categorie to specify your Bean or "No Change" if you would like to take the same category you specified when you register the old bean.
No Change
Insert your EMail address here. This email address is also used as your login string if you want to delete or update your bean.
EMail address:
Please enter your Password
Update Bean
Reset
100%

Abbildung 3.8: Der Update einer JavaBean

- die neue Kategorie, unter der die neue JavaBean auf dem Meta-Server abgelegt werden soll; wird hier keine Kategorie ausgewählt, so wird die neue JavaBean in der selben Kategorie abgelegt, unter der auch die alte JavaBean auf dem Meta-Server abgelegt war.
- die e-mail-Adresse des KleinAnbieters, die als Login für das Update einer JavaBean benutzt wird.
- das Passwort, welches sich der KleinAnbieter beim Registrieren seiner JavaBean ausgedacht hat.

Nachdem der KleinAnbieter seine Daten in die entsprechenden Felder des Update-Formulars eingegeben hat, hat er entweder durch drücken des RESET-Knopfes die Möglichkeit die Felder wieder auf den Anfangswert zurück zu setzen oder er schickt seinen Updateauftrag durch drücken des UPDATE-Knopfes ab. An dieser Stelle beginnt die eigentliche Arbeit des Update-Servlets. Es liest die Werte der Formularfelder aus und überprüft ihren Inhalt auf Richtigkeit. Zunächst wird dann versucht den Eintrag der alten JavaBean in der internen Datenbank zu finden. Im Erfolgsfall wird anschliessend je nachdem, was der KleinAnbieter gewählt hatte entweder die neue Bean von der alten oder von der neuen URL heruntergeladen und somit analog zur Registrierung verfahren. Falls der KleinAnbieter die neue JavaBean in eine andere Kategorie eingeordnet haben möchte,

so wird die neue JavaBean unter der neuen Kategorie auf dem Meta-Server abgelegt. Die alte JavaBean wird auf dem Meta-Server gelöscht (siehe Löschen einer eigenen JavaBean) und die Einträge aus der internen Datenbank entfernt.

Tritt irgendwo ein Fehler auf, so wird dem KleinAnbieter das Update-Formular erneut angezeigt und zusätzlich oben noch eine Fehlermeldung, die die Fehlerursache erläutert, mit ausgegeben. Falls das Update erfolgreich war, so bekommt der KleinAnbieter eine neue HTML-Seite vorgesetzt, auf der er über das erfolgreiche Update seiner JavaBean in der BeanDL in Kenntnis gesetzt wird und er nun verschiedene Möglichkeiten hat, weiter zu surfen.

3.5.3.4 Löschen einer JavaBean

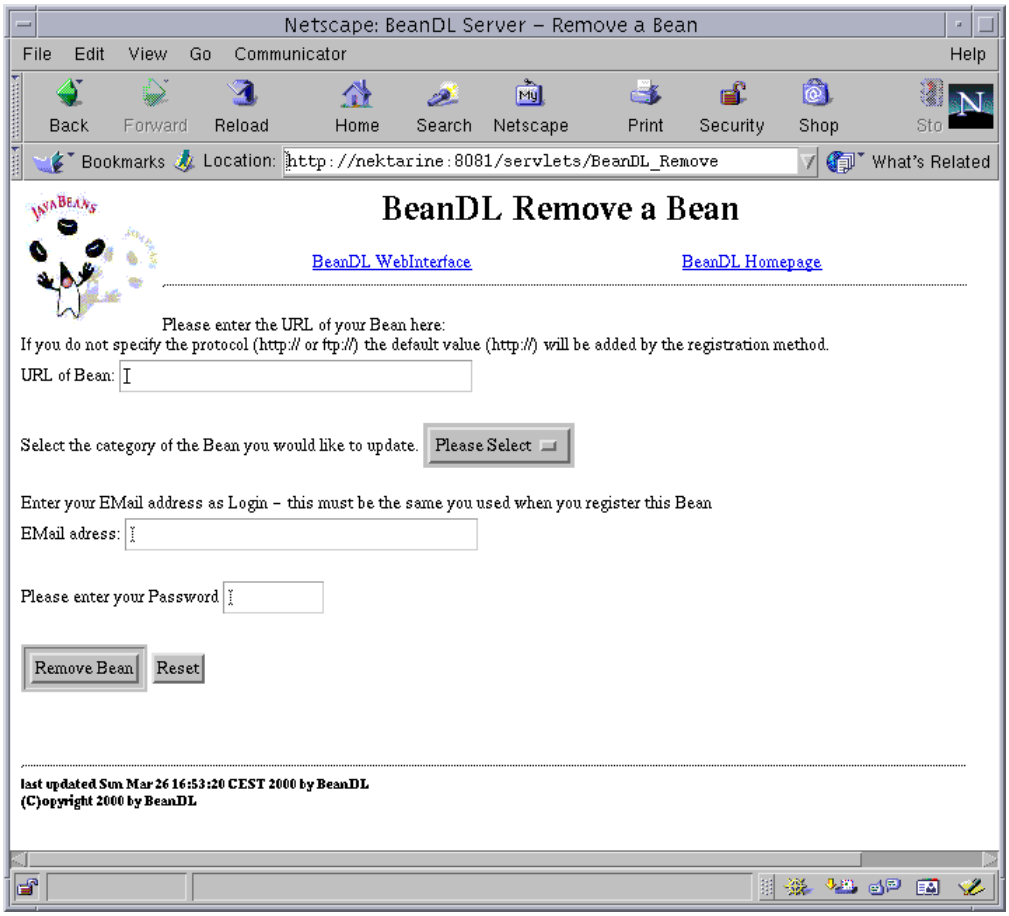
Netscape: BeanDL Server – Remove a Bean
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Sto
Bookmarks Location: http://nektarine:8081/servlets/BeanDL_Remove What's Related
BeanDL Remove a Bean
BeanDL WebInterface
BeanDL Homepage
Please enter the URL of your Bean here:
If you do not specify the protocol (http:// or ftp://) the default value (http://) will be added by the registration method.
URL of Bean:
Select the category of the Bean you would like to update.
Please Select
Enter your EMail address as Login – this must be the same you used when you register this Bean
EMail adress:
Please enter your Password
Remove Bean
Reset
last updated Sun Mar 26 16:53:20 CEST 2000 by BeanDL
(C)opyright 2000 by BeanDL

Abbildung 3.9: Das Löschen einer JavaBean

Für das Löschen seiner eigenen JavaBeans vom Meta-Server steht dem KleinAnbieter ebenfalls wieder eine HTML-Seite mit einem Formular (Abb. 3.9) zur Verfügung. Über den Link "Remove page" auf dem WebInterface gelangt er dort hin.

Der KleinAnbieter kann nur seine eigenen JavaBeans vom Meta-Server entfernen, was durch eine Abfrage seines Logins und Passwortes, sowie der URL und der Kategorie sichergestellt wird.

Zum Entfernen seiner eigenen JavaBean füllt der KleinAnbieter also in dem Remove-Formular die folgenden Felder aus:

- die URL, wo die alte JavaBean zum Upload abgelegt war. Die URL kann sowohl mit http:// als auch mit ftp:// beginnen. Falls kein Protokoll angegeben wurde, so wird automatisch http:// hinzugefügt.
- die Kategorie, unter der die JavaBean auf dem Meta-Server abgelegt wurde.
- die e-mail-Adresse des KleinAnbieters, die als Login für das Löschen seiner JavaBean benutzt wird.
- das Passwort, welches sich der KleinAnbieter beim Registrieren seiner JavaBean ausgedacht hat.

Nachdem der KleinAnbieter seine Daten in die entsprechenden Felder des Remove-Formulars eingegeben hat, hat er entweder durch drücken des RESET-Knopfes die Möglichkeit die Felder wieder auf den Anfangswert zurück zu setzen oder er schickt seinen Löschauftrag durch drücken des "REMOVE a Bean"-Knopfes ab. An dieser Stelle beginnt die eigentliche Arbeit des Remove-Servlets. Es liest die Werte der Formularfelder aus und überprüft ihren Inhalt auf Richtigkeit. Zunächst wird dann versucht den Eintrag der alten Bean in der internen Datenbank zu finden. Im Erfolgsfall werden die Einträge in der internen Datenbank gelöscht und die JavaBean auf dem Meta-Server gelöscht, indem die entsprechende Datei aus dem File-System gelöscht wird.

Tritt irgendwo ein Fehler auf, so wird dem KleinAnbieter das Remove-Formular erneut angezeigt und zusätzlich oben noch eine Fehlermeldung, die die Fehlerursache erläutert, mit ausgegeben. Falls das Löschen erfolgreich war, so bekommt der KleinAnbieter eine neue HTML-Seite vorgesetzt, auf der er über das erfolgreiche Löschen seiner JavaBean aus der BeanDL in Kenntnis gesetzt wird und er nun verschiedene Möglichkeiten hat, weiter zu surfen.

3.5.3.5 abschließende Anmerkungen

Mit dem BeanDL-Webinterface wird dem KleinAnbieter und dem Nutzer der NewBeanBox eine Schnittstelle zur Verfügung gestellt, auf die sicherlich nicht verzichtet werden sollte. Die Funktionalitäten zum Registrieren eigener neuer JavaBeans, zum Updaten von eigenen bereits registrierten JavaBeans und zum Löschen von eigenen JavaBeans, bietet dem KleinAnbieter ein umfassendes Werkzeug, um seine JavaBeans auf dem Meta-Server verwalten zu können.

Die Download-Seite bietet sowohl dem interessierten Nutzer der BeanBox die Möglichkeit, sich unsere netzwerkwerkfähige neue Version der BeanBox, die NewBeanBox, herunter zu laden, alsauch dem Anbieter von neuen JavaBeans die Möglichkeit, sich die Software für den Bean-Server herunter zu laden und diesen bei sich zu installieren und somit seine JavaBeans der Allgemeinheit zugänglich zu machen. Insgesamt dient die Download-Seite der Verbreitung der NewBeanBox und des BeanServers und somit auch der Verbreitung dieser neuen Technologie, der verteilten digitalen Beans-Bibliothek.

3.6 Abschließende Anmerkungen

Mit dem Programmieren der WWW-Schnittstellen für den KleinAnbieter haben wir uns natürlich erst beschäftigt, als der BeanServer und die NewBeanBox schon in einer fortgeschrittenen Programmierphase waren, weil ja dieser Teil nicht ganz so relevant für das Projekt BeanDL angesehen wurden, sondern mehr Wert auf die Nutzerseite gelegt werden sollte. Trotzdem sollte nicht auf diese Schnittstelle verzichtet werden, weil ja sonst keine KleinAnbieter, die keinen eigenen Bean-Server betreiben können, eigene JavaBeans der Allgemeinheit der BeanBox-Nutzer zur Verfügung stellen könnten.

Nachdem anfänglich nur das Registrieren neuer JavaBeans als Funktionalität für die WWW-Schnittstelle vorgesehen war, wurde die Funktionalität noch um das Updaten und Löschen von JavaBeans erweitert, was sicherlich eher von Vorteil sein dürfte, sonst würden möglicherweise eine Vielzahl gleicher JavaBeans in den verschiedensten Versionen auf dem Meta-Server abgelegt werden, oder JavaBeans die versehentlich in die falsche Kategorie eingeordnet wurden in mehreren Kategorien abgelegt werden, was den freien Plattenspeicher auf die Dauer ziemlich dezimieren würde und somit später eventuell dazu führen könnte, dass neue JavaBeans nicht mehr registriert werden können, da kein Speicherplatz mehr zur Verfügung steht. Ausserdem hatte ein KleinAnbieter keine Chance gehabt, seine eigene JavaBean der Öffentlichkeit wieder vorzuenthalten, wenn er sie einmal registriert hätte.

Insgesamt war das Arbeiten mit den Servlets auch eine gewisse Herausforderung, da wir ja zunächst mit “Internal Server Errors” überschüttet wurde, ohne uns diese erklären zu können, bis wir endlich wussten, wohin unsere Systemausgaben und Fehlermeldungen geschrieben wurden.

Das Layout der Formulare könnte sicherlich noch ein wenig verbessert werden, denn schließlich sind die Geschmäcker ja verschieden. Vielleicht gibt es auch noch das Eine oder Andere an der Art und Weise der Datenerfassung zu ändern, aber im Grossen und Ganzen ist die Funktionalität mit dieser gerade aktiven Version der Servlets gegeben.

Kommentare positiver wie negativer Art sind uns herzlich willkommen. Einfach eine E-Mail an eines der Gruppenmitglieder - oder an alle (Adressen befinden sich auf der BeanDL-Homepage).

Kapitel 4

Ergebnisse der Framework-Gruppe

Maik Hoeft,
Michael Malachinski,
Daniel Pawlowski,
Werner Willms

4.1 Aufgaben-Beschreibung

4.1.1 Die gestellte Aufgabe

Im Rahmen der gestellten Aufgabe soll ein allgemeines Modell entwickelt werden, das das Angebot, die Bestellung, die Abrechnung und die Bezahlung digitaler Objekte unterstützt. Die konkrete Umsetzung des entwickelten Modells soll anschließend durch eine prototypische Implementierung erfolgen.
Da es sich um ein allgemeines Modell handelt, sollen bezüglich der Art der digitalen Objekte keinerlei Einschränkungen gemacht werden. Als typische digitale Objekte, die angeboten werden können, seien jedoch Texte, Bücher, Musik, Software etc. genannt. Durch das Framework sollen weder die Zustellung der Objekte zum Käufer noch deren Benutzung durch den Käufer berücksichtigt bzw. implementiert werden.
Für das Angebot digitaler Objekte soll der Anbieter beispielsweise in der Lage sein, den angebotenen Objekten individuelle Preise zu geben sowie Rabatte, Sonderangebote und Preisnachlässe zuzuweisen. Letztere können z.B. von der Nutzungsform und dem Käufer bzw. der Käufergruppe abhängig sein. Bei der Bestellung der digitalen Objekte soll der Käufer zwischen verschiedenen Nutzungsformen der Ware (Gleitlizenz, Dauerlizenz, Campuslizenz etc.) wählen können. Für erworbenene Objekte erhält er dann die entsprechende Lizenz. Die Bezahlung der Objekte soll über verschiedene elektronische Zahlungsverfahren wie Ecash oder SET möglich sein. Insbesondere soll die Verwendung guthabenbasierter Verfahren durch die Verwaltung von Zahlungskonten durch das Framework unterstützt werden.

Grundlegende Aufgabe des zu entwickelnden Frameworks ist es, alle Daten, die für Angebot, Bestellung und Abrechnung digitaler Objekte erforderlich sind, sowie die Beziehungen zwischen diesen Daten zu verwalten. Dies umfasst unter anderem die Verwaltung von Waren, Angeboten zu Waren, Käufern und Käufergruppen, Konten, Rabatten, Bestellungen und erworbenen Lizenzen.
Weitere Aufgabe ist es, die Funktionstauglichkeit des entwickelten Modells durch eine beispielhafte Implementierung zu demonstrieren. Dazu sollen zwei Programme entwickelt werden: je eines für den Anbieter digitaler Waren und eines für den potentiellen Käufer der Waren.

4.1.2 Vorgehensweise

Die Entwicklung des Frameworks einschließlich der prototypischen Implementierung gliedert sich in folgende Teilschritte (in Klammern sind die jeweils geplanten Fertigstellungstermine der einzelnen Schritte angegeben):

- Aufgabenanalyse (01.01.2000)
- Entwurf des Datenmodells (01.01.2000)
- Entwurf graphischer Zugriffstools (01.01.2000)
- Implementierung des Datenmodells und der Schnittstellen (01.01.2000)
- Implementierung der Zugriffstools (01.01.2000)

Ziel der *Aufgabenanalyse* ist es, die an das Framework gestellten Anforderungen für Angebot, Bestellung, Abrechnung und Bezahlung digitaler Objekte zu konkretisieren. Das Resultat dieser Analysephase ist eine genaue Anforderungsdefinition sowie eine erste Festlegung der zu verwaltenden Datenobjekte.
Aufgrund dieser Anforderungsdefinition wird anschließend das *Datenmodell* als Entity-Relationship-Diagramm entworfen. Darin sind alle durch das Framework zu verwaltenden Daten und deren Beziehungen untereinander definiert.
Der *Entwurf graphischer Zugriffstools* umfasst den Entwurf der Benutzungsoberfläche für die prototypische Implementierung des Frameworks. Dabei wird das Groblayout der Ein-/Ausgabemasken von Käufer- und Verkäufersoftware entwickelt. Außerdem werden in diesem Entwicklungsschritt Ablaufdiagramme für typische Aktionen der Benutzer entworfen.
Im Entwicklungsschritt *Implementierung des Datenmodells und der Schnittstellen* wird das Datenmodell auf eine Datenbank übertragen und die Schnittstelle zum Zugriff auf die Datenbank implementiert.
Als letzter Schritt ist die *Implementierung der Zugriffstools* geplant, wobei es sich hierbei um eine prototypische, also nicht notwendigerweise vollständige, Realisierung handeln soll.

4.2 Analyse

4.2.1 Anforderungsdefinition

Bevor mit dem Entwurf des Frameworks begonnen werden konnte, musste zunächst überprüft werden, welche Funktionalität der Framework bieten sollte. Um dies zu erreichen, wurde eine Anforderungsdefinition erstellt, die Klarheit bringen sollte, wie die einzelnen Akteure des Frameworks miteinander agierten. Um hierbei nicht den Überblick zu verlieren und eine gewisse Struktur zu erhalten, wurden verschiedene Sichtweisen des Frameworks benutzt. Zu jeder Sichtweise wurde dann untersucht, welche Merkmale und Funktionalitäten ihr zugeordnet werden können. Dabei kamen folgende Ergebnisse zustande.

- *Sichtweise des Käufers:*
  - Es gibt Käufer
  - Käufer können ein Login haben
  - Wenn Kunden ein Login haben, haben sie auch ein Passwort
  - Es gibt Käufergruppen
  - Käufer können Käufergruppen angehören
  - Käufer können Waren kaufen
  - Käufer können Waren auswählen
  - Käufer können Waren bestellen
  - Käufer können Waren für Käufergruppen kaufen
  - Käufer haben kein, ein oder mehrere Konten
  - Käufergruppen haben kein, ein, oder mehrere Konten

  - Käufer können Lizenzen zugeordnet werden
  - Käufergruppen können Lizenzen zugeordnet werden

- *Sichtweise der Waren:*

  - Es gibt Waren
  - Waren können zu Bundles zusammengefasst werden
  - Bundles werden wie Waren gehandhabt
  - Waren können der Ordnung halber Warengruppen angehören
  - Waren werden Käufern angeboten
  - Waren werden Käufergruppen angeboten
  - Waren haben einen festen Grundpreis
  - Waren haben gegenüber Käufergruppen einen Rabatt
  - Waren werden Lizenzen zugeordnet
  - Waren mit unterschiedlichen Lizenzen sind unterschiedliche Waren
  - Eine Ware hat eine Menge an Lizenzen

- *Sichtweise der Warengruppen:*

  - Es gibt Warengruppen (als Ordnungssystem)
  - Warengruppen können Warengruppen angehören

- *Sichtweise der Rabatte:*

  - Es gibt Rabatte
  - Rabatte gibt es in Abhängigkeit von
    * Lizenzformen
    * Gruppengröße
    * Gruppenart
    * Warenmenge
    * Warenart
    * Warengruppenart
    * Sonderangeboten (zeitlich begrenzte Rabatte)
  - Überschneidungen bei Rabattabhängigkeiten sind möglich

- *Sichtweise der Lizenzen:*

  - Es gibt Lizenzen
  - Es gibt Campuslizenzen
  - Es gibt Einzellizenzen
  - Es gibt Gleitlizenzen
  - Es gibt den Dauererwerb (Lizenz ohne zeitliche Beschränkung)

  - Es gibt Abonnements (Lizenzen, die zeitlich begrenzt sind, sich aber automatisch verlängern)
  - Es gibt Kurzzeitlizenzen
  - Es gibt verschiedene Lizenzen für verschiedene Anbieter Digitaler Objekte
  - Jede Lizenz hat eine bestimmte Grunddauer

- *Sichtweise der Bestellung:*

  - Es gibt Bestellungen
  - Bestellungen haben ein Bestelldatum
  - Zu einer Bestellung gehört eine Ware
  - Zu einer Bestellung gehört die Anzahl der bestellten Ware
  - Eine Bestellung hat einen Käufer oder eine Käufergruppe
  - Zu einer Bestellung gehört ein Verkäufer
  - Bestellungen können eingesehen werden
  - Aboverlängerungen erfordern automatische Neubestellung
  - Bestellungen bleiben nach Ablauf der Lizenz eine bestimmte Zeit erhalten

- *Sichtweise des Verkäufers:*

  - Es gibt Verkäufer
  - Verkäufer bestimmen den Preis von Waren und Bundles
  - Verkäufer ordnen Waren Bundles zu
  - Verkäufer ordnen Waren den Warengruppen zu
  - Verkäufer geben Rabatte
  - Verkäufer bestimmt die Lizenzen einer Ware
  - Verkäufer nehmen Bestellungen entgegen
  - Verkäufer bestimmt Käufergruppen
  - Verkäufer trägt Überziehungskredite ein

- *Sichtweise der Konten:*

  - Es gibt Konten
  - Konten haben einen Überziehungskredit
  - Konten haben einen aktuellen Stand
  - Konten haben das Datum der letzten Kontobewegung

4.2.2 Entitätenbestimmung

Nachdem die Anforderungsdefinition fertig gestellt war, konnten aus ihr im direkten Zug die Entitäten abgeleitet werden, die eine zentrale Rolle im Framework spielen sollten. Diese waren im folgenden:

- Jeweils eine Entität *Einzelkäufer* und *Käufergruppe*, die Spezialisierungen einer allgemeinen Entität *Käufer* waren.
- Jeweils eine Entität *Einzelware* und *Bundleware*, die Spezialisierungen einer allgemeinen Entität *Ware* waren.
- Eine Entität *Warengruppen*, denen die Elemente der Entitäten *Einzelware* und *Bundleware* zugeordnet wurden.
- Eine Entität *Lizenzmodelle* in die eingetragen werden konnte, welche verschiedenen Lizenzierungsarten beim Handel mit den Waren möglich waren.
- Eine Entität *Konto*, die Kontostandsinformationen zu einem Käufer speicherte, sowie die Entitäten *Transaktion* und *Bezahlung*, die jede einzelne Kontobewegung speicherten.
- Eine Entität *Bestellung*, in der alle Bestellungen aufgeführt wurden, die von Käufern jemals getätigt wurden.
- Eine Entität *Angebote*, die verschiedene Angebote zu den im System verfügbaren Waren speicherte.
- Eine Entität *Rabatte*, in der verzeichnet war, bei welchen Kombinationen von Angeboten, Käufern, Waren etc. Rabatte zum normalen Angebotspreis gegeben wurden.

Aufgrund von datenbank-technischen Aspekten waren weitere Entitäten nötig, um die Relationen zwischen den oben angegebenen Entitäten herzustellen. Dies lag daran, dass in der Datenbank viele n-zu-m-Beziehungen zwischen den einzelnen Entitäten auftraten, die nur über Zwischenrelationen aufzulösen waren.

4.3 Entwurf

4.3.1 Designentscheidungen

Bevor die verschiedenen Komponenten des Frameworks entworfen werden konnten, musste zunächst die grobe Richtung gewählt werden, in der die einzelnen Entwürfe sich bewegen sollten. Mit dem Hintergedanken, dass der Framework eventuell im kommerziellen Bereich eingesetzt werden könnte und dadurch mitunter sehr viele digitale Objekte verwaltet müssten, entschlossen wir uns, für die Speicherung der zu verwaltenden digitalen Objekte die Datenbank-Sprache *SQL* zu verwenden. Dabei entstand das Problem, dass es viele Implementierungen von SQL-Sprachen gibt (ORACLE, MySQL etc.), die sich alle etwas im Sprachumfang unterscheiden. Um nun nicht auf einen bestimmten Anbieter von SQL-Systeme angewiesen zu sein, wurde auf spezielle Spracherweiterungen der einzelnen Anbieter verzichtet und schließlich der Sprachumfang von

Standard-SQL verwendet. Als spezielle Datenbank wurde schließlich ORACLE verwendet.
Was folgte war die Programmiersprache, in der die Schnittstelle zwischen Applikationen und SQL-Datenbank geschrieben werden musste. Hier wurde die Entscheidung getroffen, das *Java Development Kit* (JDK) in der Version 1.2 von der Firma SUN zu verwenden. Die Verbindung zwischen der SQL-Datenbank und den Java-Klassen wurde mit den SQL-Klassen der *Java Enterprise API* und der JDBC-Erweiterung der verwendeten ORACLE-Datenbank hergestellt. Zu beachten ist, dass alle Methoden, die von der Datenbank-Schnittstelle zur Verfügung gestellt werden, statisch implementiert wurden, so dass auch Applikationen in anderen Programmiersprachen auf die Schnittstelle zugreifen können.
Die GUI, die es dem Benutzer erlaubt, Modifikationen an der Datenbank vorzunehmen und Bestellabläufe durchzuführen wurde ebenfalls mit dem JDK in der Version 1.2 entwickelt. Dabei wurden zwei Programme entwickelt: das eine Programm für die Aktionen, die ein Käufer durchführen kann und das andere für den Verkäufer, der gleichzeitig Administrator der Datenbank ist.

4.3.2 Entwurf des Datenmodells

Die in der Analysephase gefundenen Entitäten mit ihren Merkmalen werden durch das Entity-Relationship-Modell in ein relationales Datenmodell übertragen. Dadurch entsteht ein Datenmodell mit insgesamt 24 Tabellen mit den in der Analysephase erkannten Relationen untereinander. Zur Beschreibung des Datenmodells werden erst die Tabellen mit ihren Attributen kurz erläutert und anschließend der Aufbau der Datenbank mit Hilfe eines ER-Diagramms des Programmes ErWin dargestellt.

Es konnte die Notwendigkeit folgender Tabellen erkannt werden:

FAngebote: In dieser Tabelle werden die Angebote eingetragen. Es können nur gültige Waren/Lizenz-Kombinationen aus FWarenzuordnung angeboten werden. Angebote können mit Valid auf gültig oder ungültig gesetzt werden, sie können aber nicht gelöscht werden.

Attributname	Domain	NOT NULL	Kommentar
AngebotNr	Number	ja	(Primary Key)
WarenzuordnungNr	Number	ja	Warenzuordnung(FK)
Name	String	ja	Name des Angebotes
Preis	Number	ja	Preis
Grunddauer	Number	nein	Dauer des Angebotes
Lizenzgrunddauer	Number	nein	Grunddauer
Valid	Number	nein	Gültig?

FBestellung: Diese Tabelle dient der Zuordnung einer Bezahlung aus FBezahlung zu einem Käufer aus FKaeufer. Durch das Ablaufdatum kann angegeben werden, wann die Bestellung gelöscht werden kann.

Attributname	Domain	NOT NULL	Kommentar
BestellungNr	Number	ja	(Primary Key)
Login	String	ja	Käuferlogin (FK)
BezahlungNr	Number	ja	Bezahlung (FK)
Datum	Date	ja	Datum der Bestellung
Ablaufdatum	Date	nein	Löschdatum der Bestellung

FBestellzuordnung: In dieser Tabelle wird eine Bestellung aus FBestellungg ein Angebot aus FAngebote zugeordnet. Mit Von und Bis kann ein Zeitraum angegeben werden, in dem die Bestellung gültig ist.

Attributname	Domain	NOT NULL	Kommentar
BestellzuordnungNr	Number	ja	(Primary Key)
AngebotNr	Number	ja	Angebot (FK)
BestellungNr	Number	ja	Bestellung (FK)
Anzahl	Number	ja	Anzahl
Von	Date	nein	Gültig ab Datum
Bis	Date	nein	Gültig bis Datum

FBezahlung: Zu einer Transaktion aus FTransaktion kann die Bezahlungsart eingetragen werden. Weiter kann hier eingetragen werden, ob die Bezahlung erfolgt ist.

Attributname	Domain	NOT NULL	Kommentar
BezahlungNr	Number	ja	(Primary Key)
TransaktionNr	Number	ja	Transaktion (FK)
Bezahlungsart	String	ja	Art der Bezahlung
Erfolgt	Number	nein	Zahlung erfolgt?

FBundleware: Die Tabelle der Bundlewaren ist eine Spezialisierung von FWaren. Einer Bundleware wird noch eine Warengruppe aus FWarengruppen zugeordnet.

Attributname	Domain	NOT NULL	Kommentar
BWarenNr	Number	ja	(Primary Key)
GruppenNummer	Number	nein	Warengruppe (FK)

FBundlezuordnung: In dieser Tabelle findet die Zuordnung von Einzelwaren aus FEinzelwaren zu Bundlewaren aus FBundlewaren statt.

Attributname	Domain	NOT NULL	Kommentar
BundlezuordnungNr	Number	ja	(Primary Key)
BWarenNr	Number	ja	Bundleware (FK)
EWarenNr	Number	ja	Einzelware (FK)

FEinzelkonto: Die Tabelle für das Einzelkonto ist eine Spezialisierung der Tabelle FKonto. Ein Einzelkonto kann Teilkonto eines Gruppenkontos aus FGruppenkonto sein.

Attributname	Domain	NOT NULL	Kommentar
EinzelKontoNr	Number	ja	(PK von FKonto)
GruppenKontoNr	Number	nein	Teil von Gruppenkonto?

FEinzelware: Die Tabelle für die Einzelwaren ist eine Spezialisierung der Tabelle FWaren. Es werden der Einzelware noch eine Warengruppe aus FWarengruppen zugeordnet und eventuell eine Oberware aus FEinzelware wenn es sich um einen Teil einer Ware handelt.

Attributname	Domain	NOT NULL	Kommentar
EWarenNr	Number	ja	(Primary Key)
GruppenNummer	Number	nein	Warengruppe (FK)
OberWare	Number	nein	Teil von Ware (FK)

FEKaeufer: Die Tabelle des Einzelkäufers ist eine Spezialisierung der Tabelle FKaeufer. Hier findet sich noch als weitere Angaben zur Person die Anschrift des Käufers.

Attributname	Domain	NOT NULL	Kommentar
Login	String	ja	(PK von FKaeufer)
Strasse	String	ja	Strasse
Ort	String	ja	Wohnort
PLZ	Number	ja	Postleitzahl
Telefon	String	ja	Telefon

FGruppenkonto: Hier werden die Konten eingetragen, die Gruppenkontos sind.

Attributname	Domain	NOT NULL	Kommentar
GruppenKontoNr	Number	ja	(PK von FKonto)

FKaeufer: Die Obertabelle mit den Gemeinsamkeiten von Einzel- und Gruppenkäufern. Alle Tabellen die auf Einzel- oder Gruppenkäufer zugreifen benutzen in der Regel diese Tabelle stattdessen.

Attributname	Domain	NOT NULL	Kommentar
Login	String	ja	Benutzername (PK)
Passwort	String	ja	Benutzerpasswort
Name	String	ja	Benutzername

FKaeufergruppe: Die Tabelle der Käufergruppe ist eine Spezialisierung der Tabelle FKaeufer. Es finden sich hier weitere Angaben zur Gruppe. In dieser Tabelle wird ein Einzelkäufer aus FEKaeufer als Administrator der Gruppe eingetragen und eventuell die Gruppe einer Obergruppe aus FKaeufergruppe zugeordnet.

Attributname	Domain	NOT NULL	Kommentar
GLogin	String	ja	(PK von FKaeufer)
Login	String	ja	Administrator (FK)
Obergruppe	String	nein	Obergruppe (FK)

FKaeuferzuordnung: In dieser Tabelle findet die Zuordnung von Einzelkäufern aus FEKaeufer zu einer Gruppe aus FKaeufergruppe statt.

Attributname	Domain	NOT NULL	Kommentar
KaeuferzuordnungNr	Number	ja	(Primary Key)
GLogin	String	ja	Gruppe (FK)
ELogin	String	ja	Einzelkäufer (FK)

FKonto: Die Obertabelle mit den Gemeinsamkeiten von Einzel- und Gruppenkonten. Ein Login kann mehrere Konten haben. Auch kann das Konto, zum Beispiel bei einer Überziehung, gesperrt werden.

Attributname	Domain	NOT NULL	Kommentar
KontoNr	Number	ja	(Primary Key)
Login	String	ja	Käuferlogin (FK)
Kredit	Number	ja	Kredit
Stand	Number	ja	Saldo
Gesperrt	Number	nein	Konto gesperrt?

FLizenzmodell: In dieser Tabelle befinden sich die verschiedenen Lizenzmodelle. Diese können ein Enddatum besitzen oder auf unbestimmte Dauer gültig sein.

Attributname	Domain	NOT NULL	Kommentar
LizenzmodellNr	Number	ja	(Primary Key)
LizenzName	String	ja	Name der Lizenz
HatEndDatum	Number	ja	Ist es eine Dauerlizenz?

FLizenznutzung: In dieser Tabelle kann einem Einzelkäufer aus FEKaeufer eine Lizenzbenutzung zugeordnet werden. BisDatum gibt dabei das Datum an, bis zu dem die Lizenz für den Benutzer reserviert ist.

Attributname	Domain	NOT NULL	Kommentar
Lizenznutzungsnummer	Number	ja	(Primary Key)
Login	String	ja	Käuferlogin (FK)
BestellzuordnungNr	Number	ja	Bestellung (FK)
BisDatum	Date	ja	Enddatum

FRabatt: In dieser Tabelle kann der Rabatt für ein Angebot aus FAngebote eingetragen werden. Es ist dabei eine Kombination verschiedener Merkmale möglich.

Attributname	Domain	NOT NULL	Kommentar
RabattNr	Number	ja	(Primary Key)
Rabatt	Number	ja	Rabatt in Prozent
AngebotNr	Number	ja	Angebot (FK)
GruppenNummer	Number	nein	Warengruppe (FK)
TypenName	String	nein	Typ (FK)
Gruppengroesse	Number	nein	Grösse der Gruppe
Warenmenge	Number	nein	Warenmenge
Sonderangebot	Date	nein	Enddatum des Rabattes

FRabattzuordnung: In dieser Tabelle wird einem Käufer aus FKaeufer ein Rabatt aus FRabatt zugeordnet.

Attributname	Domain	NOT NULL	Kommentar
RabattzuordnungNr	Number	ja	(Primary Key)
Login	String	ja	Käuferlogin (FK)
RabattNr	Number	ja	Rabattnummer (FK)

FTransaktion: In dieser Tabelle können die Transaktionen auf einem Konto von FKonto mit dem Verwendungszweck eingetragen werden.

Attributname	Domain	NOT NULL	Kommentar
TransaktionNr	Number	ja	(Primary Key)
KontoNr	Number	ja	Konto (FK)
Hoehe	Number	ja	Betrag
Datum	Date	ja	Datum
Zweck	String	ja	Verwendungszweck

FTypen: Hier werden die verschiedenen Käufertypen (wie etwa Studenten etc.) eingetragen.

Attributname	Domain	NOT NULL	Kommentar
TypenName	String	ja	Name des Typs (PK)

FTypenzuordnung: In dieser Tabelle findet die Zuordnung eines Käufers aus FKaeufer zu einem Käufertyp aus FTypen statt.

Attributname	Domain	NOT NULL	Kommentar
TypenzuordnungNr	Number	ja	(Primary Key)
Login	String	ja	Käuferlogin (FK)
TypenName	String	ja	Typ (FK)

FWaren: Die Obertabelle mit den Gemeinsamkeiten von Einzel- und Bundlewaren.

Attributname	Domain	NOT NULL	Kommentar
WarenNr	Number	ja	(Primary Key)
WarenName	String	ja	Name der Ware

FWarengruppen: Die Tabelle für die Warengruppen. Es kann einer Warengruppe eine Obergruppe aus FWarengruppen zugeordnet werden um so eine hierarchische Baumstruktur zu schaffen.

Attributname	Domain	NOT NULL	Kommentar
GruppenNummer	Number	ja	(Primary Key)
GruppenName	String	ja	Gruppenname
Obergruppe	Number	nein	Teilgruppe von (FK)

FWarenzuordnung: Diese Tabelle dient der Zuordnung von Waren aus FWaren zu den verschiedenen Lizenzmodellen aus FLizenzmodell um gültige Kombinationen zu bestimmen die dann angeboten werden können.

Attributname	Domain	NOT NULL	Kommentar
WarenzuordnungNr	Number	ja	(Primary Key)
LizenzmodellNr	Number	ja	Lizenzmodell (FK)
WarenNr	Number	ja	Ware (FK)

4.3.3 Entwurf der Benutzerschnittstelle

Der Entwurf der Benutzerschnittstelle ist gegliedert in Ablaufdiagramme und Maskenentwurf. Da für Käufer und Verkäufer je ein Softwaresystem entwickelt werden soll, sind Diagramme und Entwurf wiederum nach Käufer und Verkäufer unterteilt.
Die Ablaufdiagramme wurden erstellt für die grundlegenden in der Anforderungsdefinition identifizierten Funktionen. Sie stellen für diese detailliert die zur Umsetzung notwendigen Teilaufgaben und deren Reihenfolge dar. Dadurch wird die Wirkung von Interaktionen mit der

Benutzerschnittstelle festgelegt.
Die Maskenentwürfe sind das Ergebnis der Umsetzung der geplanten Funktionalität auf die Benutzungsoberfläche. Es handelt sich hierbei um Skizzen, die prinzipiell die Präsentation der Daten und Interaktionselemente durch die zu entwickelnde Software verdeutlichen sollen.

4.3.3.1 Ablaufdiagramme

Für folgende Funktionalität der Käuferanwendung wurden Ablaufdiagramme entwickelt:

- er kann sich in das System einloggen (Abbildung 4.2)
- ist er noch kein registrierter Benutzer, so kann er sich als neuen Benutzer anlegen (Abbildung 4.4)
- er kann seine bei der Registrierung eingegebenen Daten ändern (Abbildung 4.5)
- er kann sein Konto und die Transaktionen ansehen (Abbildung 4.6)
- er kann Angebote ansehen (Abbildung 4.7)
- es können Angebote von Waren nach bestimmten Kriterien gesucht und ausgegeben werden (Abbildung 4.8)
- angezeigte Angebote können in den Warenkorb gelegt werden (Abbildung 4.9)
- die Waren im Warenkorb können bestellt werden (Abbildung 4.10)
- die Lizenzen in seinem Besitz können angezeigt werden (Abbildung 4.11)
- die Bezahlung soll auf verschiedene Arten durchgeführt werden können; ist jedoch noch nicht näher spezifiziert (Abbildung 4.12)

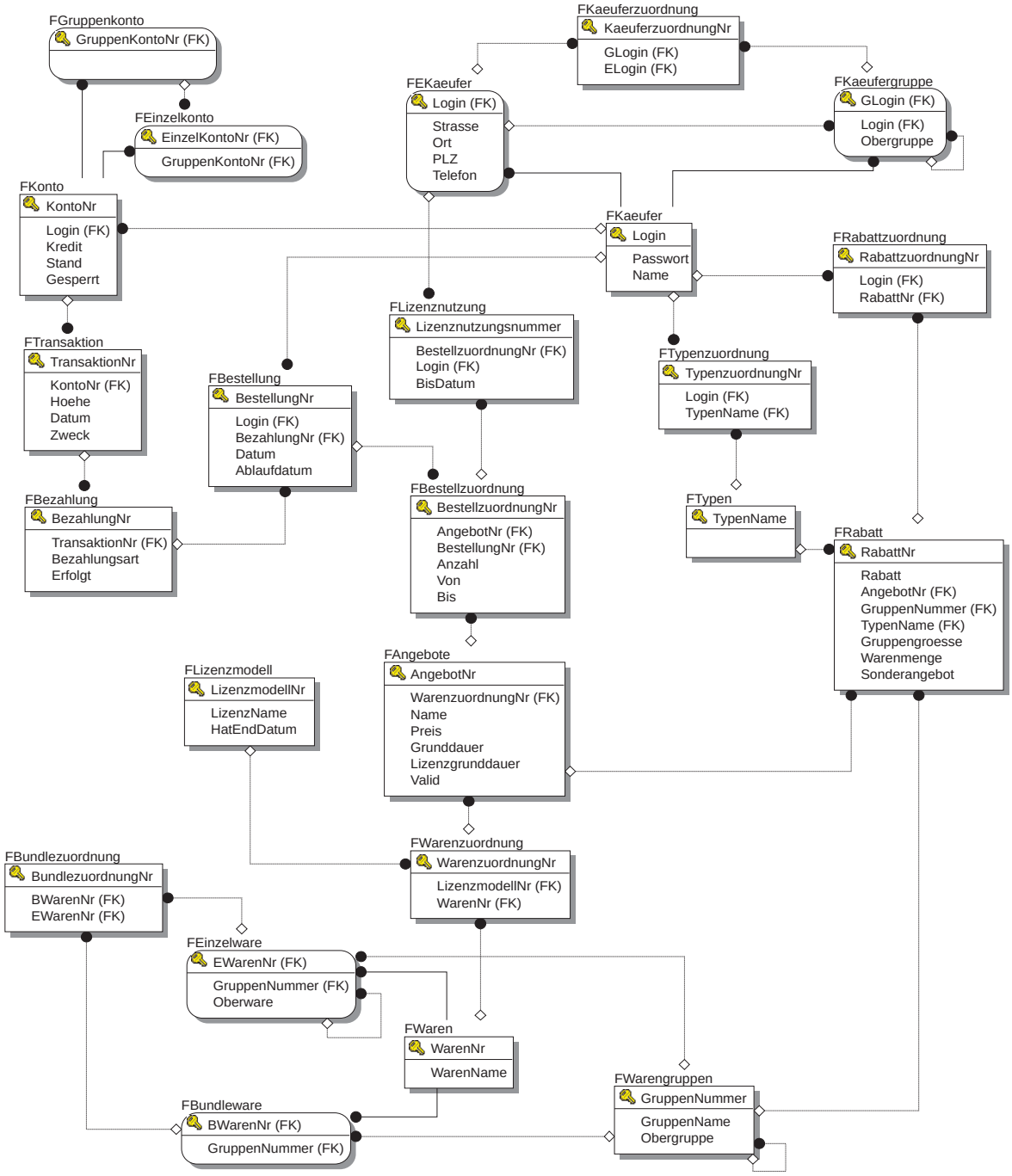

Abbildung 4.1: Das Framework-Datenbankschema

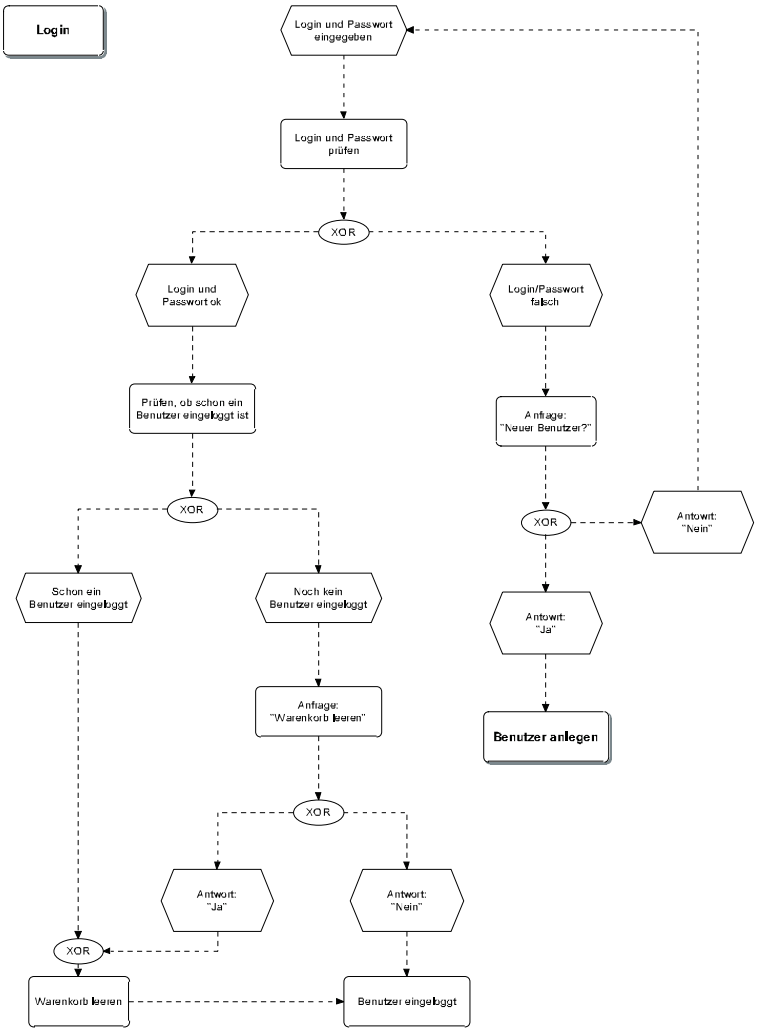

Abbildung 4.2: Login

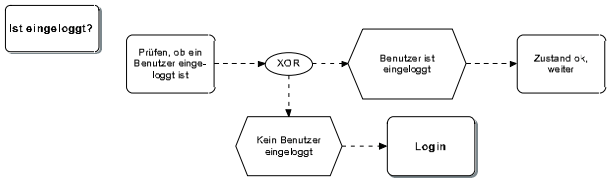

Abbildung 4.3: Ist eingeloggt

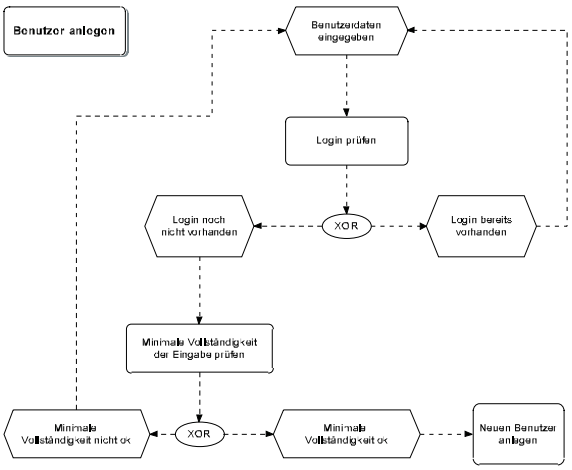

Abbildung 4.4: Benutzer anlegen

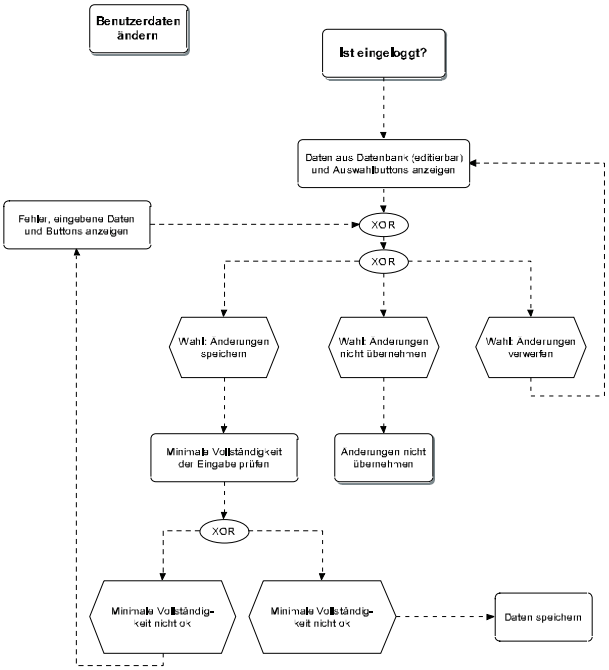

Abbildung 4.5: Benutzer ändern

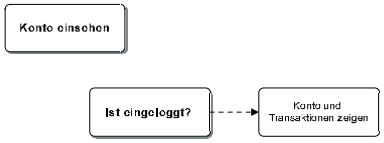

Abbildung 4.6: Konto einsehen

Abbildung 4.7: Angebote ansehen

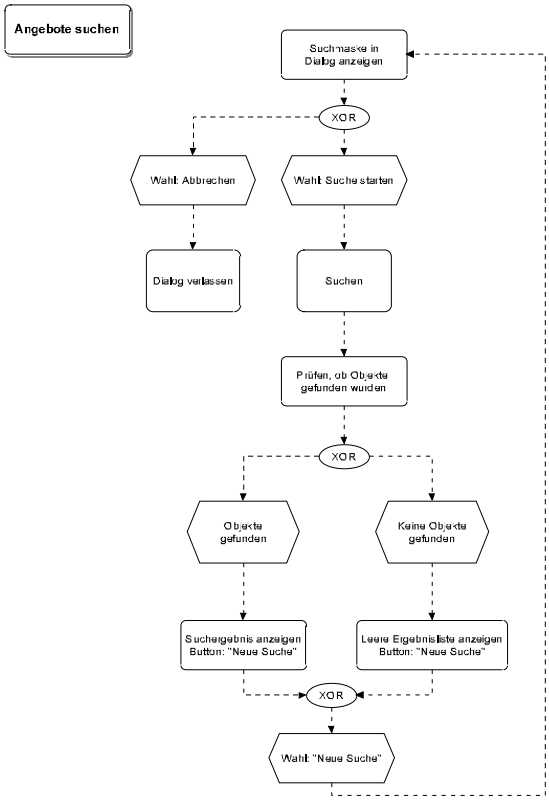

Abbildung 4.8: Angebot suchen

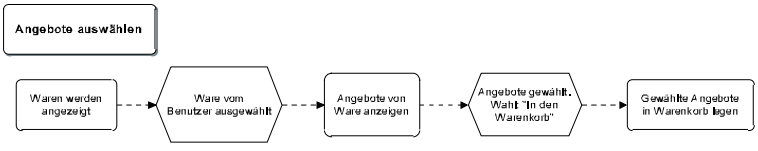

Abbildung 4.9: Angebot auswaehlen

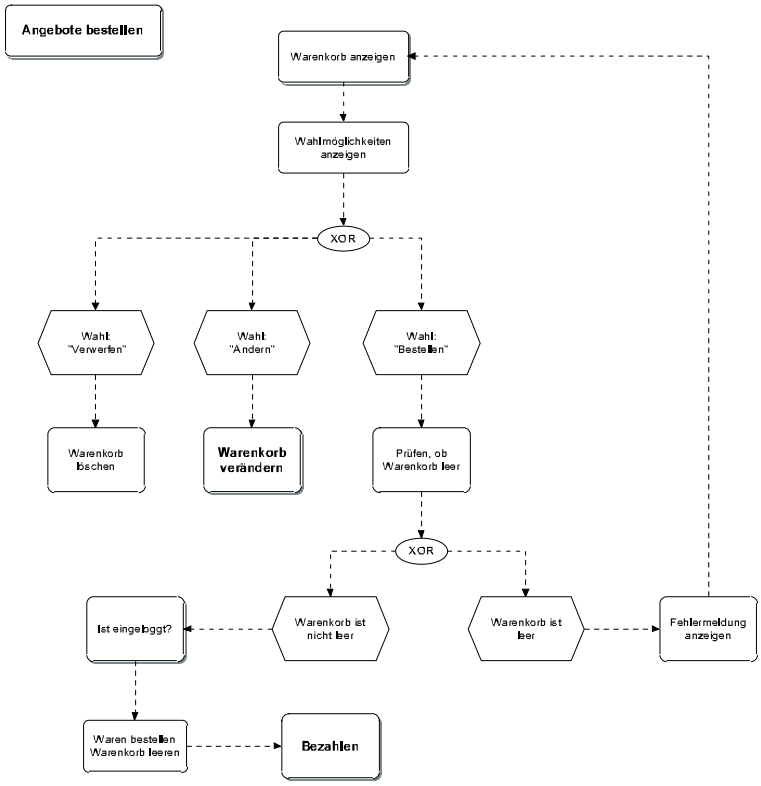

Abbildung 4.10: Angebot bestellen

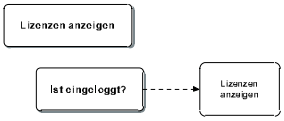

Abbildung 4.11: Lizenzen anzeigen

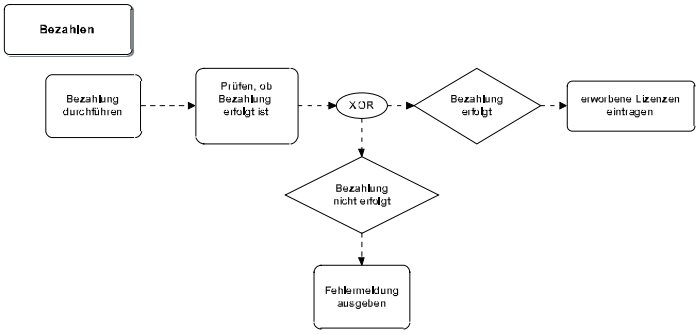

Abbildung 4.12: Bezahlen

Für den Verkäufer sind die folgenden Use Cases identifiziert und die entsprechenden FlowCharts entwickelt worden:

- er kann eine neue Ware eingeben (Abbildung 4.13)
- er kann bereits eingegebene Waren ändern (Abbildung 4.14)
- er kann ein Bundle, d.h. eine Sammlung von Waren zu einer neuen Ware, anlegen (Abbildung 4.15)
- Angebote von Waren können angelegt werden (Abbildung 4.16)
- er kann Warengruppen löschen (Abbildung 4.17)

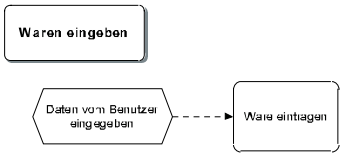

Abbildung 4.13: Ware eingeben

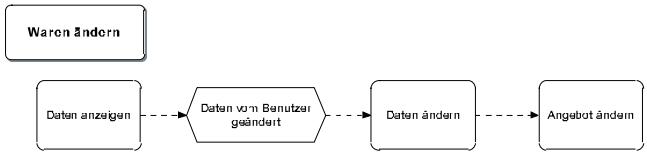

Abbildung 4.14: Ware ändern

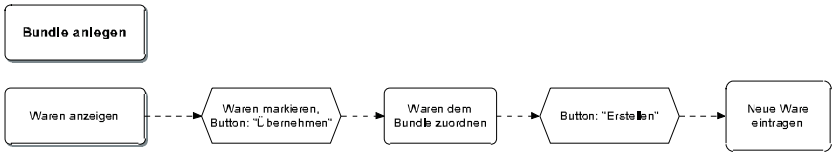

Abbildung 4.15: Bundle anlegen

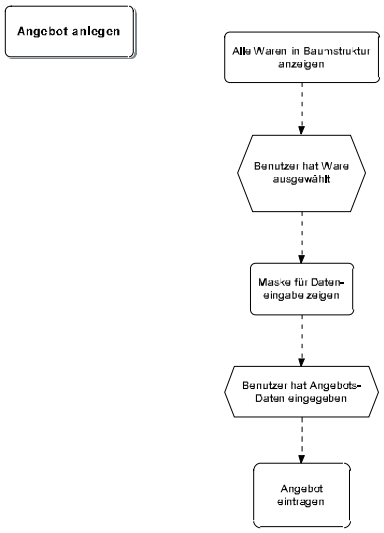

Abbildung 4.16: Angebot anlegen

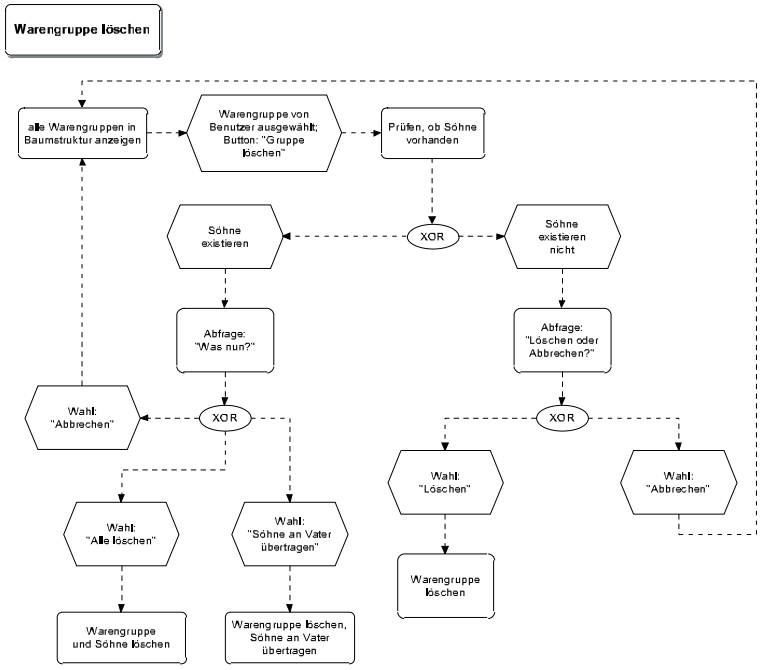

Abbildung 4.17: Warengruppe löschen

4.3.3.2 Maskenentwurf

Für die Realisierung der Applikation für den Käufer sind folgende Maskenskizzen entstanden:

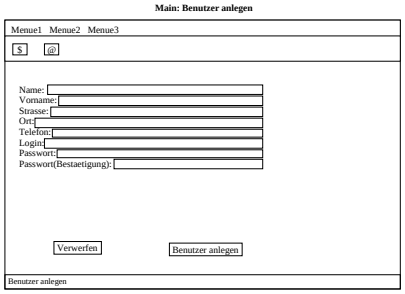

Abbildung 4.18: Benutzer Anlegen

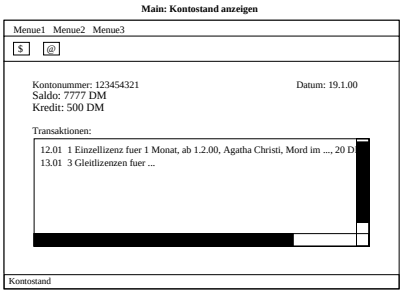

Abbildung 4.19: Kontostand anzeigen

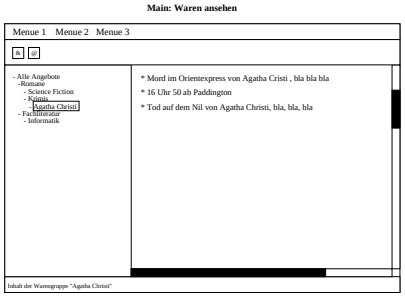
Main: Waren ansehen
Menue 1 Menue 2 Menue 3
- Alle Angebote
-Romane
- Science Fiction
- Krimis
Agatha Christi
- Fachliteratur
- Informatik
* Mord im Orientexpress von Agatha Cristi , bla bla bla
* 16 Uhr 50 ab Paddington
* Tod auf dem Nil von Agatha Christi, bla, bla, bla
Inhalt der Warengruppe "Agatha Christi"

Abbildung 4.20: Waren ansehen

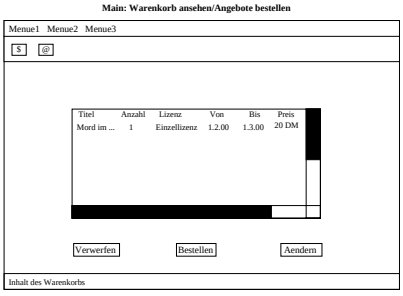

Abbildung 4.21: Warenkorb ansehen

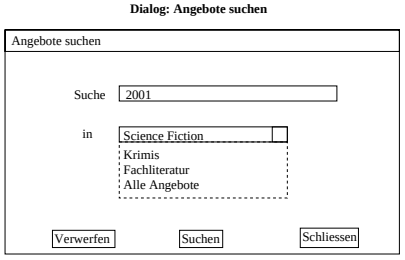

Abbildung 4.22: Angebote suchen

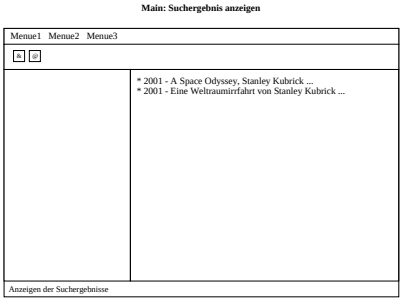
Main: Suchergebnis anzeigen
Menue1 Menue2 Menue3
* 2001 - A Space Odyssey, Stanley Kubrick ...
* 2001 - Eine Weltraumirrfahrt von Stanley Kubrick ...
Anzeigen der Suchergebnisse

Abbildung 4.23: Suchergebnisse anzeigen

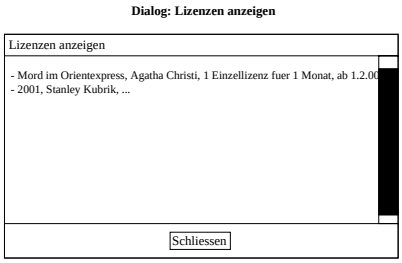
Dialog: Lizenzen anzeigen
Lizenzen anzeigen
- Mord im Orientexpress, Agatha Christi, 1 Einzellizenz fuer 1 Monat, ab 1.2.00
- 2001, Stanley Kubrik, ...
Schliessen

Abbildung 4.24: Lizenzen anzeigen

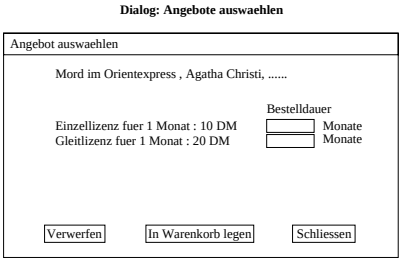

Abbildung 4.25: Angebote auswählen

Folgende Masken wurden für das Programm für den Verkäufer entworfen:

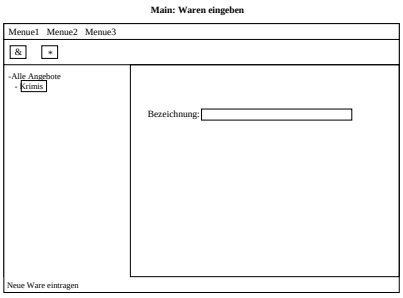

Abbildung 4.26: Waren eingeben

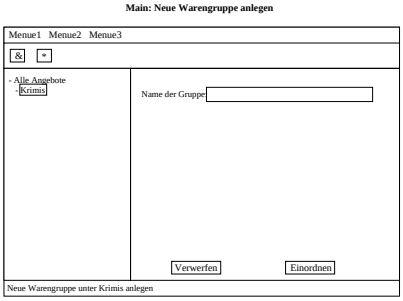

Abbildung 4.27: Warengruppe anlegen

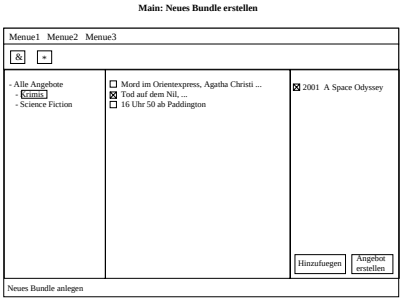

Abbildung 4.28: Waren zu Bundle zuordnen

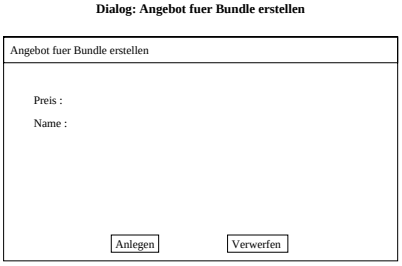

Abbildung 4.29: Bundle anlegen

4.4 Implementierung

4.4.1 Schnittstellendokumentation

4.4.1.1 Übersicht

Die Schnittstellen des Frameworks für das Angebot, die Bestellung, die Abrechnung und die Bezahlung von digitalen Objekten sind statische Methoden der Klasse FDBSchnittstelle und in der folgenden Dokumentation nach Zugehörigkeit zu den Entitäten geordnet. Da es bei einigen Schnittstellen zu Mehrdeutigkeiten kommen kann, sind an entsprechenden Stellen Verweise

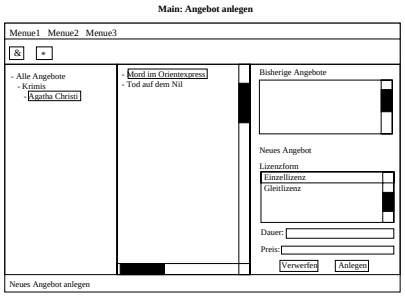

Abbildung 4.30: Angebot anlegen

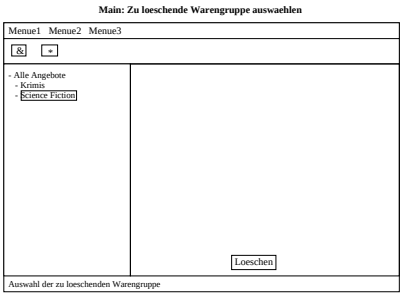

Abbildung 4.31: WarengruppeLoeschen

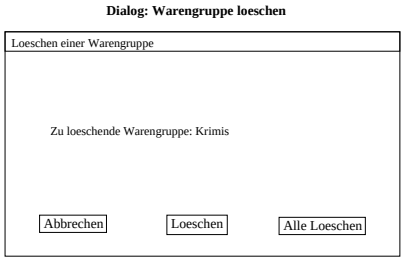

Abbildung 4.32: WarengruppeLoeschenDialog

gesetzt. Die Schnittstelle "GibAlleKontonummernVonKaeufer" ist der Entität "Kaeufer" zugeordnet und unter "Konto" ist ein Verweis auf "Kaeufer" zu finden. Die Entitäten und die in ihnen enthaltenen Schnittstellen sind alphabetisch geordnet.
Die Klassen Angebot, EinzelKaeufer, Kaeufer, Lizenz, Lizenzmodell, Transaktion, Ware und Warengruppe bestehen aus den sie kennzeichnenden Attributen und haben außer den Konstruktoren größtenteils nur Methoden, die die jeweiligen Attributwerte zurückliefern. Diese sind von der Form get*Attributname*() (wobei der Attributname mit einem Großbuchstaben beginnt) und werden in dieser Dokumentation nicht weiter beschrieben. Hingegen werden die Attribute und Konstruktoren aufgelistet und wo es notwendig ist spezielle Methoden dokumentiert.

4.4.1.2 FDBSchnittstelle

- Angebot
  - boolean AendereAngebotGueltig (int AngebotNr, boolean Gueltig)
    - *Beschreibung:* Ändert die Gültigkeit eines Angebotes
    - *Parameter AngebotNr:* Bezeichnet die ID-Nummer eines Angebotes
    - *Parameter Gueltig:* Legt fest, ob das entsprechende Angebot gültig (true) oder nicht gültig (false) sein soll
    - *Rückgabewert:* Liefert True, falls die Änderung vorgenommen wurde, False sonst
  - Angebot GibAngebot (int Angebotsnummer)
    - *Beschreibung:* Liefert das komplette Angebot zur Angebotsnummer
    - *Parameter Angebotsnummer:* Bezeichnet die Angebotsnummer
    - *Rückgabewert:* Falls die Angebotsnummer existiert, wird das entsprechende Angebots-Objekt zurückgeliefert, Null sonst

  - Vector GibAngeboteVonWare (int Warennummer)
    * *Beschreibung:* Liefert alle Angebote zu einer Ware
    * *Parameter Warennummer:* Bezeichnet die Warennummer
    * *Rückgabewert:* Ein Vektor von allen Angeboten, die zu der Ware gehören
  - int NeuesAngebot (int Warennummer, int Lizenzmodellnummer, String Name, double Preis, int Lizenzgrunddauer)
    * *Beschreibung:* Legt ein neues Angebot an, falls eine Zuordnung von Warennummer zu Lizenzmodellnummer bereits existiert
    * *Parameter Warennummer:* Bezeichnet die Ware, für die ein neues Angebot angelegt werden soll
    * *Parameter Lizenzmodellnummer:* Bezeichnet das Lizenzmodell, das für das Angebot der Ware gelten soll
    * *Parameter Name:* Der Name des Angebotes
    * *Parameter Preis:* Der Preis des Angebotes
    * *Parameter Lizenzgrunddauer:* Die Dauer, auf die sich der Preis bezieht
    * *Rückgabewert:* Falls das Angebot angelegt wurde, wird die neue Angebotsnummer zurückgegeben, -1 sonst

- Bestellung
  - boolean NeueBestellung (String Login, int Bezahlungsnummer, String Datum, String Ablaufdatum)
    * *Beschreibung:* Legt eine neue Bestellung an, wenn Login und die Bezahlungsnummer existieren
    * *Parameter Login:* Das Käuferlogin
    * *Parameter Bezahlungsnummer:* Die Nummer einer Bezahlung
    * *Parameter Datum:* Das Datum im Format dd-mmm-jj
    * *Parameter Ablaufdatum:* Ein Ablaufdatum der Bestellung im Format dd-mmm-jj
    * *Rückgabewert:* True, falls die Bestellung angelegt wurde, False sonst
  - boolean NeueBestellzuordnung (int Angebot, int Bestellungsnummer, int Anzahl, String Von, String Bis)
    * *Beschreibung:* Legt eine neue Bestellzuordnung an, falls Angebot und Bestellungsnummer existieren
    * *Parameter Angebot:* Die Angebotsnummer
    * *Parameter Bestellungsnummer:* Die Bestellungsnummer
    * *Parameter Anzahl:* Die Anzahl der Angebote in der Bestellung
    * *Parameter Von:* Anfangsdatum der Bestellzuordnung im Format dd-mmm-jj
    * *Parameter Bis:* Enddatum der Bestellzuordnung im Format dd-mmm-jj
    * *Rückgabewert:* True, falls die Bestellzuordnung angelegt wurde, False sonst

- Bezahlung
  - boolean AendereBezahlungErfolgt (int Transaktionsnummer, boolean Erfolgt)
    * *Beschreibung:* Zeichnet eine Bezahlung in Abhängigkeit von "Erfolgt" als erfolgt oder nicht erfolgt aus

    - *Parameter Transaktionsnummer:* Die Nummer der Transaktion auf die sich die Bezahlung bezieht
    - *Parameter Erfolgt:* True, falls die Bezahlung erfolgt ist, False sonst
    - *Rückgabewert:* True, falls die Bezahlung entsprechend gekennzeichnet wurde, False sonst
  - boolean NeueBezahlung (int Transaktionsnummer, String Art, int Erfolgt)
    - *Beschreibung:* Legt eine neue Bezahlung an, falls die Transaktion existiert und noch nicht in Bezahlung auftaucht
    - *Parameter Transaktionsnummer:* Die Nummer der Transaktion, auf die sich die neue Bezahlung beziehen soll
    - *Parameter Art:* Die Art der Bezahlung
    - *Parameter Erfolgt:* 0, falls die Bezahlung noch nicht erfolgt ist, 1 falls die Bezahlung erfolgt ist
    - *Rückgabewert:* True, falls die neue Bezahlung angelegt wurde, False sonst

- Käufer
  - boolean AendereKaeuferAdresse (String Login, String Strasse, String Ort, int Plz, String Telefon)
    - *Beschreibung:* Ordnet einem Käufer eine neue Adresse zu
    - *Parameter Login:* Das Login des Käufers
    - *Parameter Strasse:* Die Straße des Käufers
    - *Parameter Ort:* Der Ort
    - *Parameter Plz:* Die Postleitzahl
    - *Parameter Telefon:* Die Telefonnummer
    - *Rückgabewert:* True, falls neue Adresse angelegt wurde, False sonst
  - boolean AendereKaeuferObergruppe (String Login, String Obergruppenlogin)
    - *Beschreibung:* Ändert die Obergruppe des Gruppenkäufers wenn Login und Obergruppe existieren
    - *Parameter Login:* Das Login des Gruppenkäufers
    - *Parameter Obergruppenlogin:* Das Login einer Käufergruppe, leerer String für keine Käufergruppe
    - *Rückgabewert:* True, falls die Zuordnung erfolgreich war, False sonst
  - Vector GibAlleEinzelKaeufer ()
    - *Beschreibung:* Liefert alle existierenden Einzelkäufer zurück
    - *Rückgabewert:* Vektor von Einzelkäufer-Objekten
  - Vector GibAlleKontonummernVonKaeufer (String Login)
    - *Beschreibung:* Liefert alle Kontonummern eines Käufers, egal ob Einzelkonto oder Gruppenkonto
    - *Parameter Login:* Das Login des Käufers
    - *Rückgabewert:* Vektor von Integer, die für die Kontonummern stehen
  - EinzelKaeufer GibEinzelKaeuferZuLogin (String Login)
    - *Beschreibung:* Liefert ein Einzelkäufer-Objekt zu einem Login

    - *Parameter Login:* Das Login eines Käufers
    - *Rückgabewert:* Das entsprechende Einzelkäufer-Objekt falls vorhanden, Null sonst
- Vector GibEinzelKontonummernVonKaeufer (String Login)
    - *Beschreibung:* Liefert alle Kontonummern eines Käufers, die nicht Teil eines Gruppenkontos sind
    - *Parameter Login:* Das Login des Käufers
    - *Rückgabewert:* Vektor von Integer, die die Kontonummern darstellen
- Vector GibGruppenKontonummernVonKaeufer (String Login)
    - *Beschreibung:* Liefert alle Kontonummern eines Käufers, die Teil eines Gruppenkontos sind
    - *Parameter Login:* Das Login eines Käufer
    - *Rückgabewert:* Vector von Integer, die die Kontonummern darstellen
- Kaeufer GibKaeuferZuLogin (String Login)
    - *Beschreibung:* Liefert ein Kaeufer-Objekt zu einem Login
    - *Parameter Login:* Das Login des Käufers
    - *Rückgabewert:* Ein Kaeufer-Objekt, falls das Login vorhanden ist, Null sonst
- Vector GibLizenzenVonBenutzer (String Login)
    - *Beschreibung:* Liefert alle Lizenz-Objekte eines Käufers
    - *Parameter Login:* Das Login des Käufers
    - *Rückgabewert:* Vektor von Lizenz-Objekten
- String Kaeufername (String Login)
    - *Beschreibung:* Liefert den Käufernamen zu einem Login
    - *Parameter Login:* Das Käuferlogin
    - *Rückgabewert:* Der Name des Käufers
- boolean KaeuferZuGruppe (String KaeuferLogin, String GruppenLogin)
    - *Beschreibung:* Ordnet einer Käufergruppe einen neuen Käufer zu
    - *Parameter KaeuferLogin:* Das Käuferlogin
    - *Parameter GruppenLogin:* Das Käufergruppenlogin
    - *Rückgabewert:* True, falls der Käufer der Gruppe zugeordnet wurde, False sonst
- boolean LoginZuTyp (String Login, String Typ)
    - *Beschreibung:* Ordnet einem Käufer einen Typ zu, falls Käufer und Typ existieren
    - *Parameter Login:* Das Käuferlogin
    - *Parameter Typ:* Der Typ
    - *Rückgabewert:* True, falls Zuordnung angelegt wurde, False sonst
- boolean NeuenEinzelkaeuferAnlegen (String Login, String Passwort, String Name, String Strasse, String Ort, int Plz, String Telefon)
    - *Beschreibung:* Legt einen neuen Käufer an
    - *Parameter Login:* Das Login
    - *Parameter Passwort:* Das Passwortß

    - *Parameter Name:* Der Name
    - *Parameter Strasse:* Die Straße
    - *Parameter Ort:* Der Ort
    - *Parameter Plz:* Die Postleitzahl
    - *Parameter Telefon:* Die Telefonnummer
    - *Rückgabewert:* True, falls der neue Käufer angelegt wurde, False sonst
  - boolean NeuenGruppenkaeuferAnlegen (String GruppenkaeuferLogin, String Passwort, String Name, String AdminLogin, String ObergruppenLogin)
    - *Beschreibung:* Legt eine neue Käufergruppe an und trägt den Gruppenadmin in die Käuferzuordnung ein
    - *Parameter GruppenkaeuferLogin:* Das Login der Käufergruppe
    - *Parameter Passwort:* Das Passwort der Käufergruppe
    - *Parameter Name:* Der Name der Käufergruppe
    - *Parameter AdminLogin:* Das Login des Administrators der Gruppe
    - *Parameter ObergruppenLogin:* Das Login einer Käufergruppe, die Obergruppe der neuen Käufergruppe sein soll
    - *Rückgabewert:* True, falls die Käufergruppe angelegt wurde, False sonst

- Konto
  - boolean AendereKontoGesperrt (int Kontonummer, boolean Gesperrt)
    - *Beschreibung:* Ändert die Sperrung eines Kontos
    - *Parameter Kontonummer:* Die Kontonummer
    - *Parameter Gesperrt:* Gibt an, ob das Konto gesperrt werden soll oder nicht
    - *Rückgabewert:* True, falls die Sperrung des Kontos geändert wurde, False sonst
  - boolean AendereKontoKredit (int Kontonummer, double Kredit)
    - *Beschreibung:* Ändert den Kontokredit
    - *Parameter Kontonummer:* Die Kontonummer
    - *Parameter Kredit:* Die Höhe des neuen Kontokredits
    - *Rückgabewert:* True, falls der Kontokredit geändert wurde, False sonst
  - boolean AendereKontoStand (int Kontonummer, double Stand)
    - *Beschreibung:* Ändert den Kontostand eines Kontos
    - *Parameter Kontonummer:* Die Kontonummer
    - *Parameter Stand:* Der neue Stand des Kontos
    - *Rückgabewert:* True, falls der Kontostand geändert wurde, False sonst
  - Vector GibAlleKontonummernVonKaeufer (String Login) siehe Käufer
  - Vector GibEinzelKontonummernVonKaeufer (String Login) siehe Käufer
  - Vector GibGruppenKontonummernVonKaeufer (String Login) siehe Käufer
  - int GibKreditVonKonto (int Kontonummer)
    - *Beschreibung:* Liefert den Kredit eines Kontos
    - *Parameter Kontonummer:* Die Kontonummer des Kontos
    - *Rückgabewert:* Der Kredit für das Konto, -1 falls das Konto nicht existiert

  - double GibSaldoVonKonto (int Kontonummer)
    * *Beschreibung:* Liefert das Guthaben eines Kontos
    * *Parameter Kontonummer:* Die Kontonummer
    * *Rückgabewert:* Das Guthaben, 0.0 falls das Konto nicht existiert
  - Vector GibTransaktionsnummernVonKonto (int Kontonummer)
    * *Beschreibung:* Liefert die Transaktionsnummern zu einem Konto
    * *Parameter Kontonummer:* Die Kontonummer
    * *Rückgabewert:* Vektor von Integer, die für die Transaktionsnummern stehen
  - boolean NeuesKonto (String Login, double Kredit, double Stand, int Gesperrt, int Gruppenkonto)
    * *Beschreibung:* Legt ein neues Konto an. Je nachdem ob es sich beim Login um eine Gruppe oder einen Einzelkäufer handelt, wird ein Gruppen- oder Einzelkonto eingerichtet
    * *Parameter Login:* Das Login des Käufers, für den ein Konto eingerichtet werden soll
    * *Parameter Kredit:* Der Kredit des Kontos
    * *Parameter Stand:* Der Stand des Kontos
    * *Parameter Gesperrt:* 0 steht für nicht gesperrt, 1 für gesperrt
    * *Parameter Gruppenkonto:* -1, falls das Konto nicht Teil eines Gruppenkontos sein soll
    * *Rückgabewert:* True, falls neues Konto angelegt wurde, False sonst

- Lizenz
  - Vector GibLizenzenVonBenutzer (String Login) siehe Käufer
  - Vector GibLizenzmodelleVonWare (int Warennummer) siehe Ware
  - boolean WareZuLizenz (int Warennummer, int Lizenzmodellnummer) Warennummer) siehe Ware
  - Vector Lizenzmodelle ()
    * *Beschreibung:* Liefert alle eingetragenen Lizenzmodelle zurück
    * *Rückgabewert:* Vektor von Lizenzmodell-Objekten
  - boolean LoescheLizenznutzung (int Lizenznutzungsnummer)
    * *Beschreibung:* Löscht eine Lizenznutzung
    * *Parameter Lizenznutzungsnummer:* Die Nummer der Lizenznutzung
    * *Rückgabewert:* True, falls die Lizenznutzung gelöscht wurde, False sonst
  - boolean NeueLizenznutzung (String Login, int Bestellzuordnungsnummer, String Datum)
    * *Beschreibung:* Legt eine neue Lizenznutzung an
    * *Parameter Login:* Login des Käufers, für den die Lizenznutzung gilt
    * *Parameter Bestellzuordnungsnummer:* Die Zuordnungsnummer zu einer Bestellung
    * *Parameter Datum:* Das Datum bis wann die Lizenz belegt ist im Format dd-mmm-jj

    - *Rückgabewert:* True, falls die neue Lizenznutzung angelegt wurde, False sonst
  - boolean NeuesLizenzmodell (String Name, boolean Befristet)
    - *Beschreibung:* Legt ein neues Lizenzmodell an, falls der Name nicht bereits existiert
    - *Parameter Name:* Der Name des neuen Lizenzmodells
    - *Parameter Befristet:* True falls das Lizenzmodell befristet ist, False falls es eine Dauerlizenz ist
    - *Rückgabewert:* True, falls das neue Modell angelegt wurde, False sonst
  - boolean WareZuLizenz (int Warennummer, int Lizenzmodellnummer) siehe Ware

- Passwort
  - boolean Passwortpruefen (String Login, String Passwort)
    - *Beschreibung:* Prüft, ob Login und Passwort zueinander passen
    - *Parameter Login:* Das Login eines Käufers
    - *Parameter Passwort:* Das Passwort
    - *Rückgabewert:* True, falls beide zueinander passen, False sonst

- Rabatt
  - boolean NeueRabattzuordnung (String Login, int Rabattnummer)
    - *Beschreibung:* Ordnet einem Login einen Rabatt zu, falls Login und Rabatt existieren
    - *Parameter Login:* Das Login eines Käufers
    - *Parameter Rabattnummer:* Die Rabattnummer
    - *Rückgabewert:* True, falls die Zuordnung durchgeführt wurde, False sonst
  - boolean NeuerRabatt (int Angebotsnummer, int Warengruppe, String Typ, double Rabatt, int Gruppengroesse, int Warenmenge, String Enddatum)
    - *Beschreibung:* Legt einen neuen Rabatt an, falls die Angebotsnummer existiert
    - *Parameter Angebotsnummer:* Das Angebot auf das sich der Rabatt bezieht
    - *Parameter Warengruppe:* Die Warengruppe auf die sich der Rabatt bezieht
    - *Parameter Typ:* Der Typ auf den sich der Rabatt bezieht, kann leer sein
    - *Parameter Rabatt:* Die Rabatthöhe in Prozent
    - *Parameter Gruppengröße:* Die Gruppengröße, kann leer sein
    - *Parameter Warenmenge:* Die Warenmenge, kann leer sein
    - *Parameter Enddatum:* Das Enddatum im Format dd-mmm-jj
    - *Rückgabewert:* True, falls neuer Rabatt angelegt wurde, False sonst

- Transaktion
  - String GibDatumVonTransaktion (int Transaktionsnummer)
    - *Beschreibung:* Liefert das Datum einer Transaktion
    - *Parameter Transaktionsnummer:* Die Transaktionsnummer
    - *Rückgabewert:* Liefert das Datum als String in dem Format ddmmyyyy. Existiert die Transaktionsnummer nicht, wird Null zurückgegeben

  - double GibHoeheVonTransaktion (int Transaktionsnummer)
    * *Beschreibung:* Liefert die Höhe einer Transaktion
    * *Parameter Transaktionsnummer:* Die Transaktionsnummer
    * *Rückgabewert:* Gibt die Höhe einer Transaktion zurück. Existiert keine Transaktion mit übergebener Nummer, wird Null zurückgegeben
  - Vector GibTransaktionsnummernVonKonto (int Kontonummer) siehe Konto
  - String GibZweckVonTransaktion (int Transaktionsnummer)
    * *Beschreibung:* Liefert das Zweckfeld einer Transaktion
    * *Parameter Transaktionsnummer:* Die Transaktionsnummer
    * *Rückgabewert:* Zweck als String. Existiert die Transaktion nicht, wird Null zurückgegeben
  - boolean NeueTransaktion (int Kontonummer, double Betrag, String Datum, String Zweck)
    * *Beschreibung:* Legt eine neue Transaktion an, wenn die Kontonummer existiert
    * *Parameter Kontonummer:* Die Kontonummer
    * *Parameter Betrag:* Der Betrag
    * *Parameter Datum:* Das Datum im Format dd-mmm-jj
    * *Parameter Zweck:* Der Zweck
    * *Rückgabewert:* True, falls die Transaktion angelegt wurde, False sonst

- Typ
  - boolean LoginZuTyp (String Login, String Typ) siehe Käufer
  - boolean NeuenTypAnlegen (String Typ)
    * *Beschreibung:* Testet ob der Typ bereits existiert und trägt ihn gegebenenfalls ein
    * *Parameter Typ:* Der Typname
    * *Rückgabewert:* True, falls der neue Typ angelegt wurde, False sonst

- Ware
  - boolean AendereOberware (int Warennummer, int Oberwarennummer)
    * *Beschreibung:* Ordnet einer Ware eine neue Oberware zu
    * *Parameter Warennummer:* Die Nummer der Ware, die eine neue Oberware erhalten soll
    * *Parameter Oberwarennummer:* Die Nummer der Oberware, -1 falls die Ware keine Oberware haben soll
    * *Rückgabewert:* True, falls die Zuordnung durchgeführt wurde, False sonst
  - Vector GibLizenzmodelleVonWare (int Warennummer)
    * *Beschreibung:* Liefert die Lizenzmodell-Objekte zu einer Ware
    * *Parameter Warennummer:* Die Warennummer
    * *Rückgabewert:* Ein Vektor von Lizenzmodell-Objekten der Ware
  - Ware GibWare (int Warennummer)

    - *Beschreibung:* Liefert das Ware-Objekt mit der Warennummer.
    - *Parameter Warennummer:* Die Warennummer
    - *Rückgabewert:* Das Ware-Objekt mit der Warennummer, Null falls die Warennummer nicht existiert
  - Vector GibWarenVonBundle (int Bundlenummer)
    - *Beschreibung:* Liefert die zu einem Bundle gehörenden Einzelwaren zurück
    - *Parameter Bundlenummer:* Die Bundlenummer
    - *Rückgabewert:* Vektor von Ware-Objekten
  - Vector GibWarenVonGruppe (int Gruppennummer)
    - *Beschreibung:* Gibt zu einer Gruppe die in ihr enthaltenen Waren zurück
    - *Parameter Gruppennummer:* Die Gruppennummer, deren Waren gesucht werden, -1 für die Wurzel
    - *Rückgabewert:* Vektor von Ware-Objekten
  - boolean NeueWare (String Name, int Gruppennummer, boolean Einzelware, int Oberwarennummer, Vector Waren)
    - *Beschreibung:* Legt eine neue Ware an.
    - *Parameter Name:* Der Name der neuen Ware
    - *Parameter Gruppennummer:* Die Gruppennummer, unter der die neue Ware eingeordnet werden soll
    - *Parameter Einzelware:* True falls neue Ware Einzelware ist, False sonst
    - *Parameter Oberwarennummer:* Die Nummer einer Oberware, von der die neue Ware ein Teil sein soll, -1 falls keine Oberware vorhanden
    - *Parameter Waren:* Ein Vektor von Wareobjekten, die Teil des Bundles sind, das erzeugt wird, falls Einzelware auf False gesetzt ist
    - *Rückgabewert:* True, falls die neue Ware angelegt wurde, False sonst
  - boolean WareZuBundle (int Warennummer, int Bundlenummer)
    - *Beschreibung:* Ordnet einem Bundle eine Ware zu falls die Ware nicht gleich dem Bundle ist und beide existieren
    - *Parameter Warennummer:* Die Warennummer
    - *Parameter Bundlenummer:* Die Bundlenummer
    - *Rückgabewert:* True, falls die Zuordnung stattfand, False sonst
  - boolean WareZuLizenz (int Warennummer, int Lizenzmodellnummer)
    - *Beschreibung:* Ordnet einer Ware eine Lizenz zu, falls beide existieren
    - *Parameter Warennummer:* Die Warennummer
    - *Parameter Lizenzmodellnummer:* Die Lizenzmodellnummer
    - *Rückgabewert:* True, falls zugeordnet, False sonst

- Warengruppe
  - boolean AendereObergruppe (int Gruppennummer, int Obergruppennummer)
    - *Beschreibung:* Ändert die Oberwarengruppe zu einer Warengruppe, falls beide existieren

    - *Parameter Gruppennummer:* Die Nummer der Warengruppe, die eine neue Obergruppe erhalten soll
    - *Parameter Obergruppennummer:* Die Nummer der neuen Oberwarengruppe
    - *Rückgabewert:* True, falls die Neuzuordnung geglückt ist, False sonst
- boolean AendereWareZuWarengruppe (int Warennummer, int Gruppennummer)
    - *Beschreibung:* Ordnet einer Ware eine neue Warengruppe zu
    - *Parameter Warennummer:* Die Nummer der Ware
    - *Parameter Gruppennummer:* Die Nummer der Warengruppe, -1 falls die Ware direkt an die Wurzel gehängt werden soll
    - *Rückgabewert:* True, falls Zuordnung geglückt ist, False sonst
- Vector GibAlleGruppen ()
    - *Beschreibung:* Liefert einen Vektor zurück, der alle existierenden Warengruppen-Objekte enthält
    - *Rückgabe:* Vektor von Warengruppen-Objekten
- Vector GibSoehneVonWarengruppe (int Vater)
    - *Beschreibung:* Liefert alle existierenden Warengruppen unterhalb der Warengruppe mit der Nummer "Vater"
    - *Parameter Vater:* Die Nummer der Obergruppe, deren Söhne gesucht werden, -1 wenn die Obergruppe die Wurzel sein soll
    - *Rückgabewert:* Vektor von Warengruppen-Objekten
- Vector GibWarengruppenOhneObergruppe ()
    - *Beschreibung:* Liefert alle Warengruppen, die direkt unter der Wurzel stehen
    - *Rückgabewert:* Vektor mit Warengruppen-Objekten
- Vector GibWarenVonGruppe (int Gruppennummer) siehe Ware
- boolean LoescheWarengruppe (int Warengruppennummer)
    - *Beschreibung:* Löscht eine Warengruppe und hängt die Söhne und Waren einen Knoten höher im Baum
    - *Parameter Warengruppennummer:* Die Nummer der zu löschenden Warengruppe
    - *Rückgabewert:* True, falls die Warengruppe gelöscht wurde, False sonst
- boolean NeueWarengruppe (String Gruppenname, int Obergruppennummer)
    - *Beschreibung:* Testet ob die Gruppe mit der Obergruppennummer existiert und trägt sie dann ein
    - *Parameter Gruppenname:* Name der neuen Warengruppe
    - *Parameter Obergruppennummer:* Die Obergruppe, unter der die neue Warengruppe eingefügt werden soll, -1 falls die neue Gruppe direkt unter der Wurzel eingefügt werden soll
    - *Rückgabewert:* True, falls die neue Warengruppe eingefügt werden konnte, False sonst
- Vector SucheWareInGruppe (String Warenname, int Gruppennummer)
    - *Beschreibung:* Sucht innerhalb und unterhalb der angegebenen Gruppe nach Warennamen, die den übergebenen Warennamen enthalten und gibt die entsprechenden Angebote zurück

  - *Parameter Warenname:* Der zu suchende Warenname
  - *Parameter Gruppennummer:* Die Gruppe, ab der gesucht werden soll, -1 steht für alle Gruppen
  - *Rückgabewert:* Vektor von Angeboten, die zu den gefundenen Waren gehören

4.4.1.3 Angebot

- Attribute:
  - int key
  - String name
  - int dauer
  - String lizenzForm
  - double preis
  - boolean gueltig
- Konstruktoren:
  - public Angebot(int key, String name, int dauer, String lizenzForm, double preis, boolean gueltig)
- spezielle Methoden:
  - public boolean isValid() Liefert den Wert von gueltig zurück

4.4.1.4 Einzelkaeufer

- EinzelKaeufer extends Kaeufer
- zusätzliche Attribute:
  - String strasse
  - int plz
  - String ort
  - String telefon
- Konstruktoren:
  - public EinzelKaeufer(String login, String name, String strasse, int plz, String ort, String telefon) Erstellt einen neuen Einzelkaeufer mit allen Attributen
- spezielle Methoden: keine

4.4.1.5 Kaeufer

- Attribute:
  - String login
  - String name
- Konstruktoren:
  - public Kaeufer(String login, String name)
- spezielle Methoden:
  - public boolean equals(Kaeufer kaeufer) Prüft zwei Käufer auf Gleichheit

4.4.1.6 Lizenz

- Attribute:
  - int bestellZuordnung
  - String titel
  - String lizenzart
  - String von
  - String bis
- Konstruktoren:
  - Lizenz(int bestellZuordnung, String titel, String lizenzart, String von, String bis)
- spezielle Methoden: keine

4.4.1.7 Lizenzmodell

- Attribute:
  - int key
  - String name
  - double anteil
  - boolean dauerlizenz
- Konstruktoren:
  - public Lizenzmodell(int key, String name, boolean dauerlizenz) Erzeugt ein Lizenzmodell mit Anteil von 100%
  - public Lizenzmodell(int key, String name, double anteil) Erzeugt ein Lizenzmodell ohne unendliche Dauer
  - public Lizenzmodell(int key, String name, double anteil, boolean dauerlizenz) Erzeugt ein Lizenzmodell mit allen Eigenschaften
- spezielle Methoden:
  - public boolean istDauerlizenz()

4.4.1.8 Transaktion

- Attribute:
  - String datum
  - String zweck
  - double betrag
- Konstruktoren:
  - Transaktion(String datum, String zweck, double betrag) Erzeugt eine neue Transaktion mit allen Attributen
- spezielle Methoden: keine

4.4.1.9 Ware

- Attribute:
  - String name
  - int key
- Konstruktoren:
  - public Ware(String name) Erzeugt eine neue Ware mit Primaerschluessel '-1'
  - public Ware(String name, int key) Erzeugt eine neue Ware
- spezielle Methoden: keine

4.4.1.10 Warengruppe

- Attribute:
  - int key
  - String name
- Konstruktoren:
  - public Warengruppe(String name) Erzeugt eine Warengruppe mit Primaerschluessel '-1'
  - public Warengruppe(String name, int key) Erzeugt eine Warengruppe
- spezielle Methoden: keine

4.4.2 Benutzerhandbuch des Käufers

Der Applikation für den Käufer stellt alle wichtigen Funktionalitäten zur Verfügung, die nötig sind, um Angebote zu sichten, auszuwählen, zu bestellen und schließlich zu bezahlen. Ausgewählte Angebote werden temporär im so genannten *Warenkorb* abgelegt. Eine Bestellung von Angeboten bedeutet, dass *alle* Angebote des Warenkorbs bestellt und durch eine ausgewählte Bezahlungsart bezahlt werden. Hierbei ist zu beachten, dass der Käufer in der Angebots-Menge nach Belieben stöbern und sich seinen Warenkorb zusammenstellen kann. Sobald der Käufer jedoch den Inhalt des Warenkorbs bestellen oder auf sensible Daten wie z.B. seinen Kontostand zugreifen möchte, muss er sich durch Eingabe seines Benutzernamens und Passworts dem Programm gegenüber verifizieren. Sollte dies nicht möglich sein, so bleiben ihm diese Funktionalitäten verwehrt. Wie das Programm für den Käufer im Einzelnen zu bedienen ist, wird in den folgenden Unterkapiteln gezeigt.

4.4.2.1 Anlegen und Verifizieren des Käufers

Sobald der Käufer Angebote bestellen möchte, muss er dem System bekannt sein. Dies wird erreicht, indem der Käufer den Menüpunkt *Benutzer* → *Benutzer anlegen* anwählt. Dadurch wird das in Abbildung 4.33 dargestellte Fenster angezeigt. Hier muss der Käufer Angaben zu seiner Person machen, sowie einen Benutzernamen (Login) und ein Passwort wählen. Sind alle Angaben korrekt und der Benutzername nicht bereits vergeben, wird der Käufer als neuer Kunde in der Datenbank eingetragen und ihm ein Konto zugeteilt.

Abbildung 4.33: Anlegen eines neuen Benutzers

Sollte der Käufer bei der Arbeit mit dem Programm aufgefordert sich zu verifizieren, muss er dies durch die Eingabe seines Benutzernamens (Login) und Passwortes erledigen.

4.4.2.2 Anlegen von Gruppen

Der Käufer hat die Möglichkeit, Käufergruppen anzulegen. Dazu ist es jedoch erforderlich, dass der Käufer, der das Programm aktuell benutzt, ein Einzelkäufer ist, der als Administrator (Verwalter) der Käufergruppe fungiert, und dass sich dieser bereits dem Programm gegenüber verifiziert hat. Sollte letzteres nicht der Fall sein, wird der Käufer dazu aufgefordert. Nach erfolgter Verifikation erscheint dann der in Abbildung 4.34 abgebildete Dialog zum Anlegen einer neuen Käufergruppe. Der Einzelkäufer hat die Möglichkeit, der neuen Käufergruppe einen Namen, einen Benutzernamen (Login) und ein Passwort zu vergeben. Benutzername und Passwort dürfen maximal wieder acht Zeichen lang sein, müssen aber auf jeden Fall angegeben werden. Weiterhin hat der Einzelkäufer die Möglichkeit seine neue Gruppe einer Obergruppe zuzuordnen. Hierbei ist es allerdings erforderlich, dass der Einzelkäufer auch Administrator dieser Obergruppe ist.

Abbildung 4.34: Anlegen einer Käufergruppe

4.4.2.3 Auswählen von Angeboten

Beim Start des Programms für den Käufer zunächst der in Abbildung 4.35 dargestellte Fenster angezeigt. Dieses Fenster besitzt auf der linken Seite eine Baumstruktur, in der alle verfügbaren Warengruppen dargestellt werden. Durch einen einfachen Klick mit dem Mauszeiger auf das Symbol links neben dem Gruppennamen wird eine Warengruppe geöffnet und alle Untergruppen angezeigt, bzw. die geöffnete Warengruppe wieder geschlossen. Durch einen einfachen Klick auf einen Warengruppennamen, werden in der Tabelle auf der rechten Seite des Fensters alle Waren angezeigt, die sich in der gewählten Warengruppe befinden. Durch Verwendung der *Steuerungs-Taste* (Strg), bzw. der *Hochstell-Taste* (Shift) bei der Auswahl von Warengruppen ist eine *Mehrfachauswahl*, bzw. *Bereichsauswahl* von Warengruppen möglich. Durch Doppelklick auf eine Warengruppe wird diese ebenfalls geöffnet oder geschlossen und gleichzeitig die Waren in der gewählten Gruppe angezeigt.

Die Anzeige von Angeboten zu einer Ware wird erreicht, indem in der Tabelle auf der rechten Seite des Fensters ein Doppelklick auf die gewünschte Ware durchgeführt wird. Dadurch öffnet sich das Fenster in Abbildung 4.36. Hier kann zu jedem Angebot durch Eingabe von Anfangs- und Enddatum entschieden werden, wie lange es in Anspruch genommen werden soll. Die ausgewählten Angebote können dann schließlich durch den Knopf *In den Warenkorb übernehmen* in den Warenkorb übernommen werden.

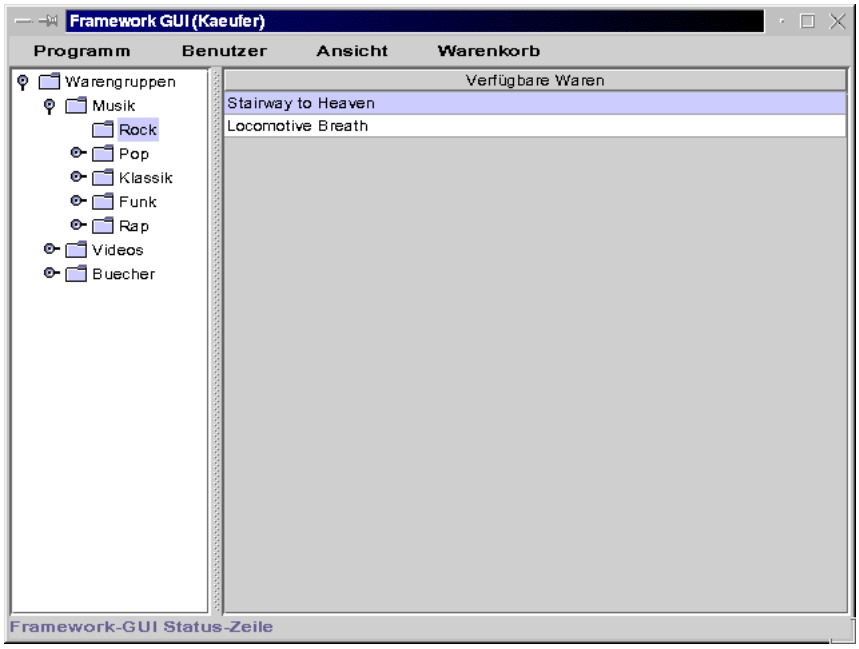

Abbildung 4.35: Auswählen von Waren

Abbildung 4.36: Auswählen von Angeboten

4.4.2.4 Bestellen von Angeboten

Der gefüllte Warenkorb kann über die Anwahl des Menüpunktes *Warenkorb* → *Warenkorb ansehen* betrachtet werden. Das erscheinende Fenster, welches in Abbildung 4.37 abgebildet ist, stellt die Angebote dar, die der Käufer später bestellen möchte. Über den Knopf *Angebote bestellen*, bzw. über den Menüpunkt *Warenkorb* → *Warenkorb bestellen* wird - einen gefüllten Warenkorb vorausgesetzt - der Dialog aufgerufen, der die Angebote bestellt. Voraussetzung ist, dass der Käufer sich beim Programm bereits verifiziert hat. Sollte dies nicht der Fall sein, wird der Käufer aufgefordert, sich zu verifizieren. Dem verifizierten Käufer präsentiert sich dann der in Abbildung 4.38 dargestellte Dialog. Hier können einerseits die gewünschte Bezahlungsart, sowie - mehrere Konten des Käufers vorausgesetzt - das Konto des Käufers gewählt werden, von dem die Bezahlung erfolgen soll. Durch den Knopf *Abschicken* wird die Bestellung und die Bezahlung, falls möglich, durchgeführt und der Warenkorb nach erfolgter Bestellung geleert. Nach erfolgter Bezahlung besitzt der Käufer Lizenzen für die Bestellten Angebote und kann diese in

Anspruch nehmen.

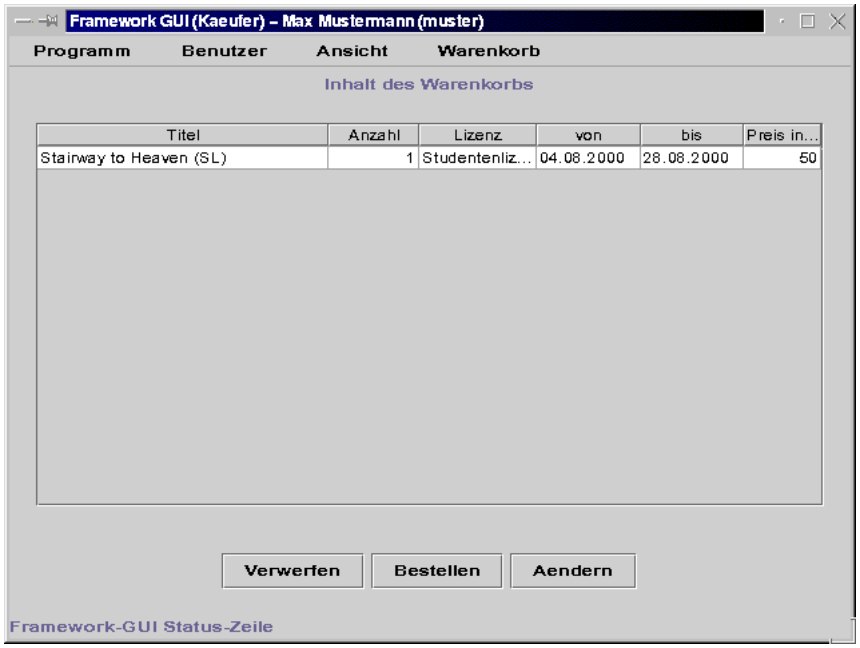

Abbildung 4.37: Ansicht des Warenkorbs

Abbildung 4.38: Dialog zum Bestellen von Angeboten

4.4.2.5 Lizenzen des Käufers anzeigen

Durch die Anwahl des Menüpunktes *Ansicht* → *Lizenzen anzeigen* wird das in Abbildung 4.39 dargestellte Fenster angezeigt. Hier werden alle gültigen Lizenzen zu Angeboten dargestellt, die der Käufer durch Bestellung *und* Bezahlung erworben hat. Angezeigt werden Informationen zum Angebot, zur Lizenzart, zum Datum des Inkrafttretens und des Endes der Lizenz. Auf die Angebote, die in diesem Fenster dargestellt werden, hat der Käufer ein Recht auf Freischaltung, also darf er diese Leistungen in Anspruch nehmen.

4.4.2.6 Kontostand des Käufers anzeigen

Jeder verifizierte Käufer besitzt mindestens ein Konto in der Datenbank. Jede Kontobewegung erfordert eine Transaktion, die mit einem Eintrag auf dem Kontoauszug einer Bank vergleichbar ist. Über den Menüpunkt *Ansicht* → *Kontostand anzeigen* kann der Käufer sich seine Konten anzeigen lassen. Das Fenster, welches dies darstellt, ist in Abbildung 4.40 abgebildet. Über eine

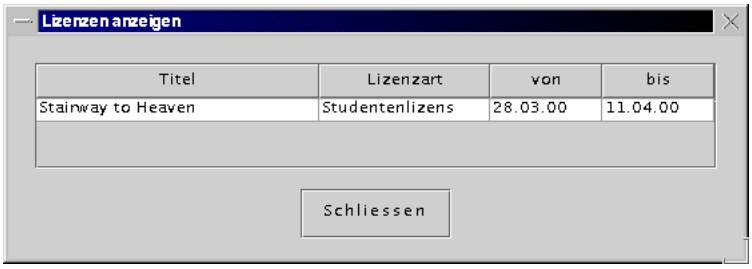

Abbildung 4.39: Anzeige der gültigen Lizenzen des Käufers

so genannte Combo-Box können die verschiedenen Konten eines Käufers ausgewählt werden. Angezeigt wird der aktuelle Kontostand und die Höhe des eingeräumten Kredits des gewählten Kontos, sowie die darauf erfolgten Transaktionen. Zu jeder Transaktion gehört ein Betrag, ein Datum und ein Verwendungszweck, anhand dessen der Käufer zurückverfolgen kann, wofür die Transaktion getätigt wurde. Sollte der Kontostand negativ sein, ist der Benutzer immer noch in der Lage, Transaktionen zu tätigen, sofern die Höhe der Transaktionen nicht die Höhe des Kredits abzüglich des *Soll* des Kontos übersteigt.

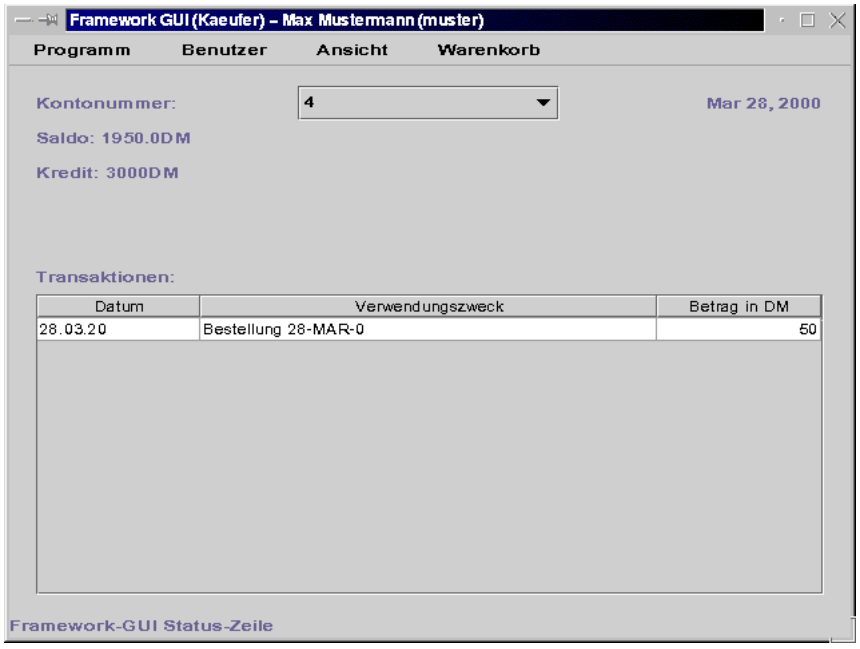

Abbildung 4.40: Anzeige des Kontostandes

4.4.3 Benutzerhandbuch der Verkäufers

Das Programm für den Verkäufer bietet Funktionalitäten, um die Artikeldatenbank aufzubauen und zu pflegen. Wie beim Programm für den Käufer werden hierbei jedoch nur die wichtigsten Funktionalitäten geboten, die nötig sind, um bequem mit der Datenbank zu arbeiten. Der Verkäufer stellt insofern also eine Art System-Administrator dar, der vollen Zugriff auf die Da-

tenbank hat.
Das System der Datenbank lässt sich wie folgt erklären. Die Datenbank besteht aus Waren, die über Angebote zu den einzelnen Waren dem Käufer präsentiert werden. Da es sich bei den Waren - wie in der Anforderung bereits beschrieben - um Digitale Objekte handelt, müssen die Waren über ein Lizenzierungs-System an den Mann (oder die Frau) gebracht werden. Hierzu ist es notwendig, dass verschiedene Lizenzmodelle zur Verfügung stehen, die den einzelnen Waren zugeordnet werden können. Zu einer Kombination aus Ware und Lizenzmodell kann dann ein Angebot erstellt werden, welches dem Käufer präsentiert werden kann. Anzumerken ist, dass Waren auch Teilwaren anderer Waren sein können (z.B. die einzelnen Kapitel eines digitalisierten Buches), und dass Waren verschiedener Art wiederum zusammengefasst werden können zu einem Bundle, was natürlich auch wieder eine Ware darstellt. Um eine gewisse Ordnung in das System der Waren zu bringen, können Warengruppen erstellt werden, in denen die Waren eingeordnet werden können.
Die Aufgabe der Grafischen Oberfläche ist es nun, den Verkäufer zu unterstützen, dieses System von Waren(gruppen), Lizenzen und Angeboten zu verwalten. Hierbei ist allerdings anzumerken, dass in der aktuellen Version der Datenbank keine Modifikationen erlaubt sind, die Elemente aus der Datenbank löschen.

4.4.3.1 Anlegen von Warengruppen

Durch die Anwahl des Menüpunktes *Ware* → *Warengruppen anlegen* gelangt der Verkäufer zu dem in Abbildung 4.41 dargestellten Fenster. Dieses Fenster ist gleichzeitig das erste Fenster, welches beim Start des Programms angezeigt wird. Auf der linken Seite kann man die bereits vorhandenen Warengruppen sehen, die in der Datenbank existieren. Durch Doppelklick auf eine Warengruppe oder mit Einfachklick des Mauszeigers auf das Symbol links neben dem Warengruppennamen öffnet sich die gewählte Warengruppe und alle ihr untergeordneten Warengruppen erscheinen. Durch Einfachklick auf eine Warengruppe wird diese markiert und kann somit als Obergruppe einer neuen Warengruppe fungieren. Sollte keine Warengruppe markiert sein, wird eine neue Warengruppe als Untergruppe der obersten Warengruppe eingetragen. In dem Eingabefeld muss der Name der Warengruppe angegeben werden, die neu erzeugt werden soll. Über den Knopf *Einordnen* wird die neue Warengruppe in der Datenbank eingefügt, der Knopf *Verwerfen* leert das Eingabefeld.

4.4.3.2 Anlegen von Waren

Waren sind der grundlegende Bestandteil der Datenbank. Waren werden immer einer Warengruppe zugeordnet, sollen Waren keiner speziellen Warengruppe zugeordnet werden, kann der Verkäufer z.B. eine Warengruppe *sonstiges* erstellen und die Waren dann dieser Warengruppe zuordnen. Waren können eingegeben werden, indem man den Menüpunkt *Ware* → *Waren anlegen*. Daraufhin erscheint das in Abbildung 4.42 abgebildete Fenster. Auf der linken Seite findet man eine Baumstruktur, in der alle verfügbaren Warengruppen dargestellt werden. Wie in der Baumstruktur navigiert wird, findet man im vorigen Kapitel. Durch markieren einer Warengruppe in der Baumstruktur erscheinen in der Mitte des Fensters die ihr zugeordneten Waren. In dem Eingabefeld muss der Name der neuen Ware eingetragen werden. Durch die Auswahl des Knopfes *Ware Eintragen* wird einer neue Ware unter dem angegebenen Namen in die Datenbank eingetragen und der gewählten Warengruppe zugeordnet. Sollte keine Warengruppe

Abbildung 4.41: Anlegen einer Warengruppe

markiert worden sein, wird die Ware der obersten Warengruppe zugeordnet. Über den Knopf *Verwerfen* kann man das Eingabefeld wieder leeren.

4.4.3.3 Lizenzmodelle anlegen

Um die in der Datenbank verfügbaren Waren dem Käufer verfügbar zu machen, ist es nötig, den einzelnen Waren Lizenzmodelle zuzuordnen. Eingegeben werden können diese, indem man den Menüpunkt *Lizenz* → *Lizenzmodelle anlegen*, woraufhin das in Abbildung 4.43 dargestellte Fenster erscheint. In der Mitte des Fensters werden tabellarisch alle derzeit in der Datenbank befindlichen Lizenzmodelle angezeigt. Sollte es sich bei einem Lizenzmodell um eine Dauerlizenz - also eine Lizenz ohne bestimmte Dauer - handeln, kann die in der rechten Spalte der Tabelle abgelesen werden. Im Eingabefeld unten im Fenster kann muss der Verkäufer einen Namen für ein neu zu erstellendes Lizenzmodell vergeben. Optional kann durch markieren des Schalters rechts neben dem Eingabefeld gewählt werden, ob es sich beim neuen Lizenzmodell um eine Dauerlizenz handelt. Durch Einfachklick auf den Knopf *Anlegen* wird das neue Lizenzmodell in der Datenbank erstellt und erscheint in der Tabelle in der Mitte des Fensters. Über den Knopf *Verwerfen* wird das Eingabefeld geleert und der Schalter rechts daneben de-selektiert.

4.4.3.4 Lizenzzuordnungen anlegen

Nachdem Waren und Lizenzmodelle eingegeben wurden, kann der Verkäufer nun die Zuordnungen der einzelnen Lizenzmodelle zu den einzelnen Waren vornehmen. Dazu ist es nötig, den Menüpunkt *Lizenz* → *Zuordnungen eintragen* auszuwählen. Daraufhin erscheint das in Abbildung 4.44 abgebildete Fenster, welches sich in drei Bereiche aufteilt. Der linke Bereich stellt

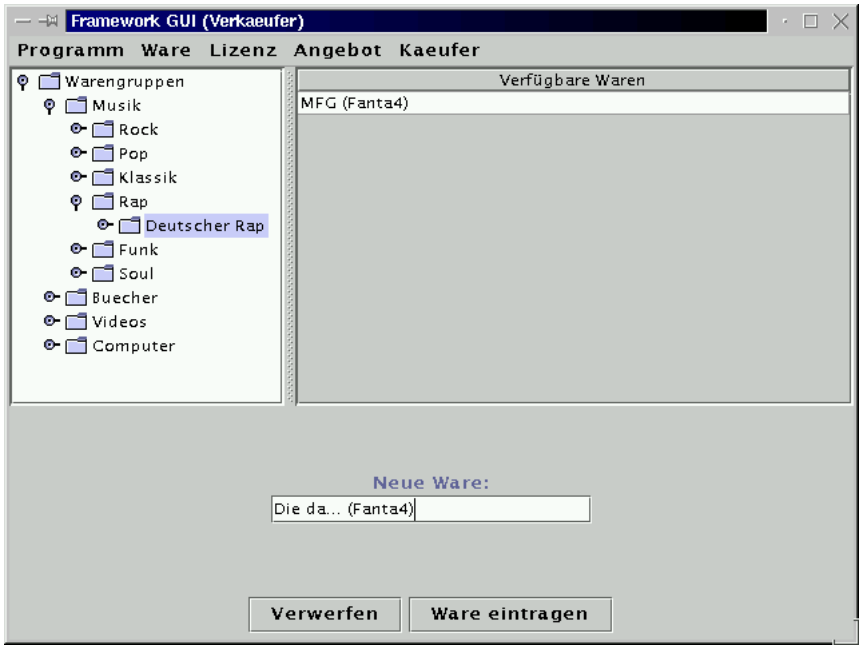

Abbildung 4.42: Anlegen einer Ware

mittels einer Baumstruktur alle Warengruppen dar, die sich in der Datenbank befinden. Durch markieren einer Warengruppe werden in der Mitte des Fensters die der Warengruppe zugeordneten Waren angezeigt. Hierbei ist es möglich, über das Festhalten der Steuerungstaste (Strg) beim Markieren mehrere Warengruppen zu wählen. Das Festhalten der Hochstelltaste (Shift) beim Markieren erlaubt, ganze Bereiche von Warengruppen zu wählen. Auf der rechten Seite des Fensters findet man die Optionen zum Erstellen einer neuen Zuordnung zwischen einer Ware und einem Lizenzmodell. Die aktuell ausgewählte Ware in der Mitte des Fensters wird oben rechts angezeigt. Daneben befindet sich eine sogenannte ComboBox, die alle in der Datenbank verfügbaren Lizenzmodelle enthält. Durch betätigen der ComboBox werden weitere Lizenzmodelle angezeigt, die gewählt werden können. Im unteren Teil der rechten Seite befinden sich zur gewählten Ware alle Zuordnungen, die sich in der Datenbank befinden. Durch Einfachklick auf den Knopf *Modell hinzufuegen* wird die gewünschte Zuordnung in die Datenbank eingetragen und schließlich im Fenster angezeigt.

4.4.3.5 Angebote anlegen

Über den Menüpunkt *Angebot* → *Angebote anlegen* können neue Angebote in die Datenbank aufgenommen werden. Dazu ist es erforderlich, dass in der Datenbank Waren, Lizenzmodelle und Zuordnungen zwischen Waren und Lizenzmodellen existieren. Das erscheinende Fenster aus Abbildung 4.45 unterteilt sich in drei große Bereiche. Auf der linken Seite und in der Mitte befindet sich eine Baumstruktur zur Anzeige der verfügbaren Warengruppen sowie die Waren in der gewählten Warengruppe. Näheres hierzu findet sich im vorherigen Kapitel. Auf der rechten Seite des Fensters stehen die Funktionalitäten zur Verfügung, die nötig sind, um Angebote zu erstellen. Durch Auswahl einer Ware in der Mitte des Fensters werden alle Informationen der Ware angezeigt. Dazu gehören alle Angebote, die bereits zu dieser Ware existieren und alle

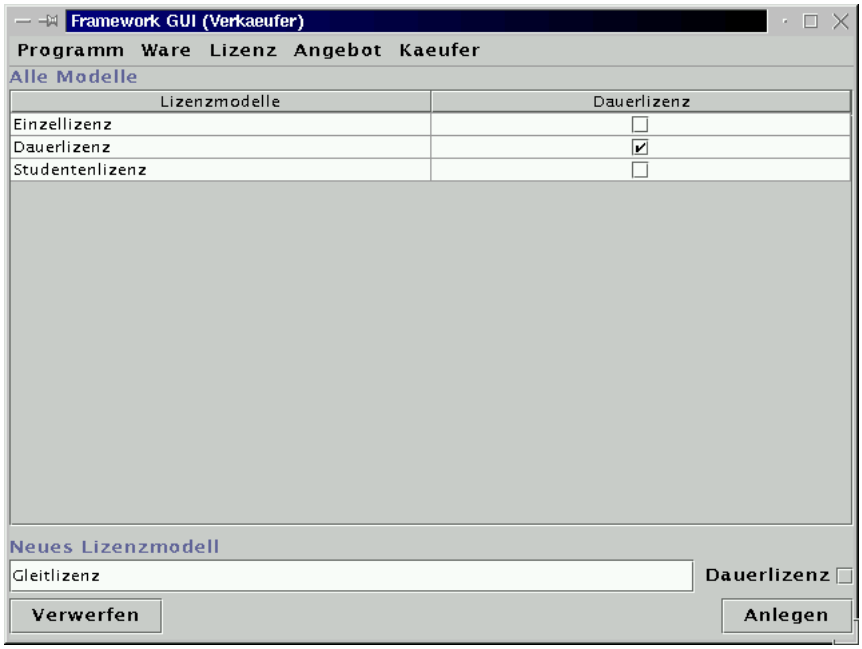

Abbildung 4.43: Anlegen eines Lizenzmodells

Lizenzmodelle, die bereits zu dieser Ware zugeordnet wurden. Um ein Angebot zu erstellen ist es nötig, zunächst ein Lizenzmodell aus der Tabelle in der Mitte der rechten Seite des Fensters zu wählen. Desweiteren muss dem Angebot ein Name gegeben werden, sowie ein Preis und einen Bezugszeitraum für diesen Preis. Dies ist notwendig, da Lizenzen zu Waren vom Käufer dynamisch erworben werden können und sich das fällige Entgelt eben aus diesem angegebenen Preis und dessen Bezugszeitraum berechnet. Sobald alle Eingaben vollständig sind, kann durch Einfachklick mit dem Mauszeiger auf den Knopf *Neues Angebot eintragen* das neue Angebot in die Datenbank übernommen werden. Dieses steht dann den Käufern zum Erwerb zur Verfügung.

4.4.3.6 Verwalten von Käufergruppen

Ebenso wie der Käufer ist auch der Verkäufer in der Lage, Käufergruppen anzulegen. Obendrein kann der Verkäufer den Käufergruppen zusätzlich neue Käufer zuordnen. Die Gruppenverwaltung kann über das Menü *Kaeufer* → *Gruppen verwalten* gestartet werden, woraufhin das in Abbildung 4.46 abgebildete Fenster erscheint. Auf der linken Seite des Fensters werden alle Gruppen angezeigt, die sich in der Datenbank befinden. Die Mitte zeigt alle Käufer, die sich bei der Datenbank registriert haben. Durch Auswahl einer der Gruppen auf der linken Seite werden rechts im Fenster alle Käufer angezeigt, die sich in der gewählten Käufergruppe befinden. Durch Einfachklick mit der Maus auf den Knopf *Kaeufer hinzufuegen* wird ein in der Mitte des Fensters markierter Kaeufer zur gewählten Gruppe hinzugefügt, vorausgesetzt er existiert nicht bereits in dieser Gruppe.

Neue Gruppen können erstellt werden, indem auf den Knopf *Neue Gruppe* einfach geklickt wird. Dadurch öffnet sich der in Abbildung 4.47 dargestellte Dialog. Die Bedienung dieses Dialogs funktioniert im Prinzip genau so, wie im äquivalenten Dialog für den Käufer. Allerdings muss der Verkäufer nun bestimmen, welcher Einzelkäufer der Administrator der Gruppe sein soll.

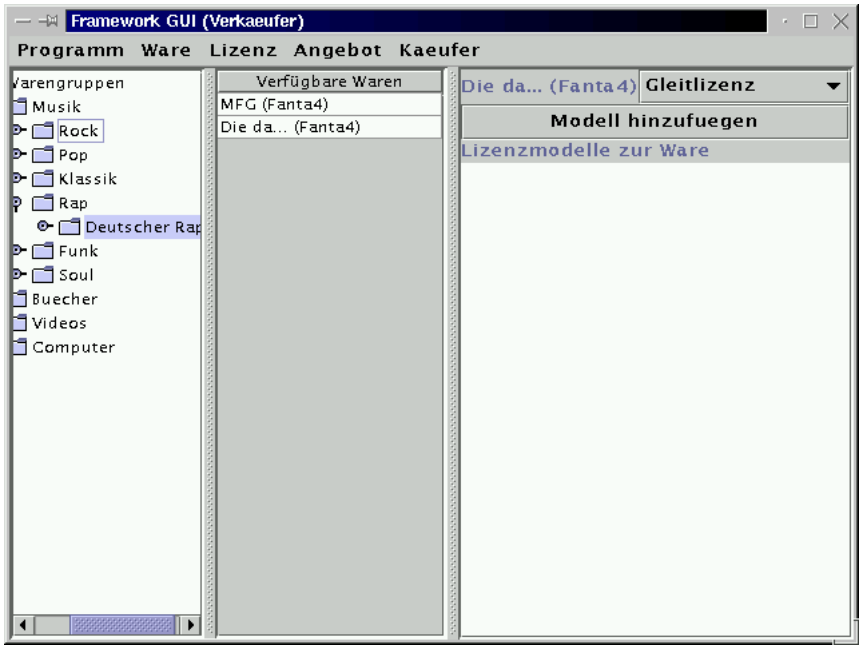

Abbildung 4.44: Zuordnen von Lizenzmodellen zu Waren

Dazu werden links im Dialog alle Einzelkäufer aufgelistet, die sich in der Datenbank registriert haben. Dieses Verfahren ist notwendig, da der Verkäufer in der Datenbank nicht als Käufer auftritt und deshalb auch nicht als Administrator einer Käufergruppe fungieren kann. Falls der Verkäufer eine eigene Gruppe haben möchte, kann er dies nur erreichen über einen eigenen Account als Einzelkäufer.

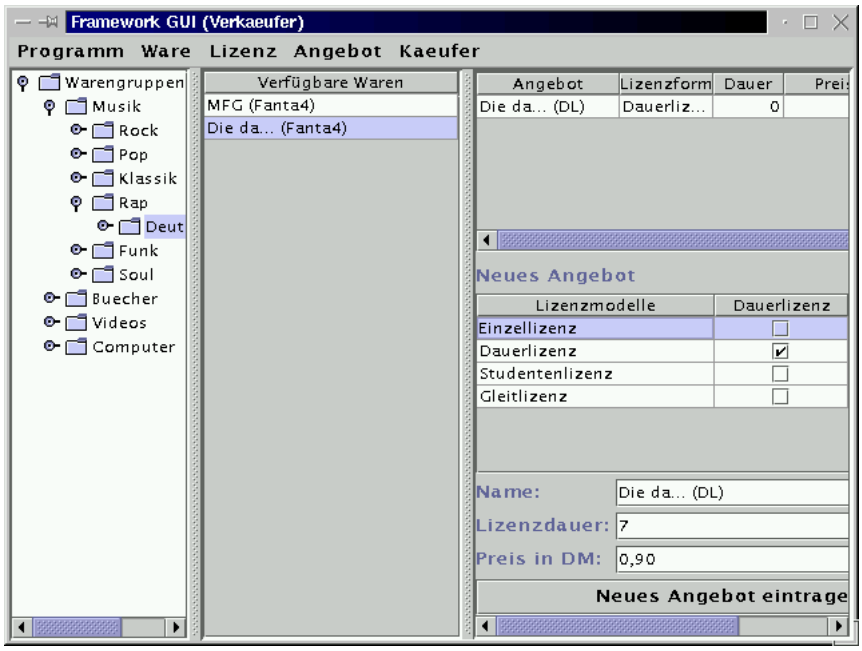

Abbildung 4.45: Erstellen von neuen Angeboten

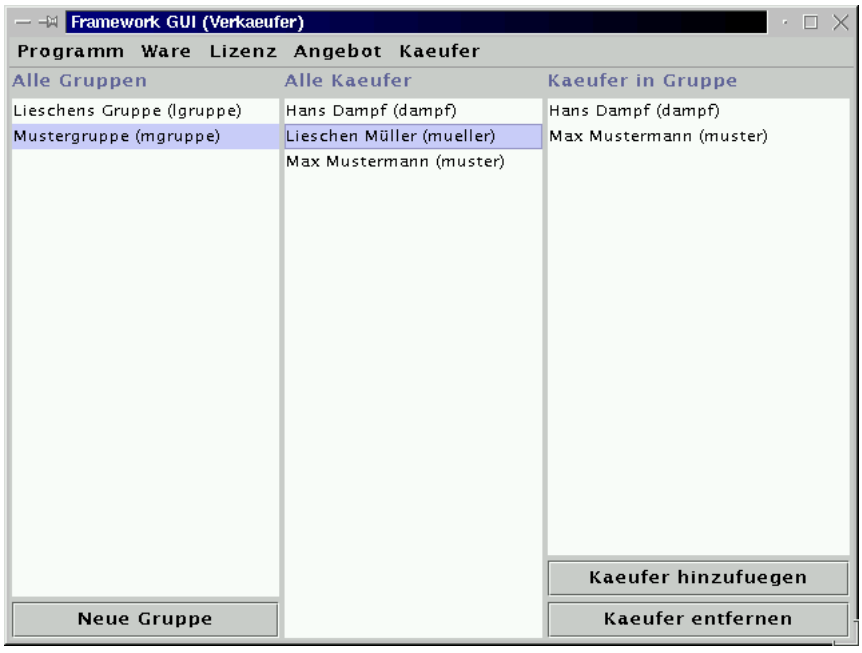

Abbildung 4.46: Hinzufügen von Käufern zu Gruppen

Abbildung 4.47: Anlegen einer Käufergruppe (Verkäufer)

Kapitel 5

Ergebnisse der Sport Digital Library

Michael Klein,
Oliver Klein,
Andre Rolfs,
Axel Wunsch

5.1 Einleitung

5.1.1 Motivation

Das vorliegende Sub-Projekt fand im Rahmen der Projektgruppe „Werkzeuge für Digitale Bibliotheken“ statt. Als erstes Teilziel innerhalb der Projektgruppe sollten die Anforderungen, Eigenschaften, Funktionsweise, Aufbau sowie die Probleme mit der Entwicklung von Digitalen Bibliotheken näher kennen gelernt werden. Damit die Teilnehmer der Projektgruppe in kleinerem Umfang praktische Erfahrungen auf diesem Gebiet sammeln konnten, beschäftigten sie sich mit der prototypischen Entwicklung eigener Digitaler Bibliotheken mit klar abgegrenzten und eingeschränkten Inhalten in einem festgelegten thematischen Kontext. Insbesondere sollten hier auch erste Erfahrungen mit Softwarewerkzeugen gesammelt werden, die die Entwicklungsarbeit unterstützen können.

5.1.2 Das Sub-Projekt Sport-DL

Diese Dokumentation behandelt das Sub-Projekt, das unter dem Titel Sport-DL lief und sich mit der Entwicklung einer Digitalen Bibliothek im sportwissenschaftlichen Anwendungsbereich beschäftigte. Für das Projekt Sport-DL gab es mit dem Institut für Sportwissenschaft der Universität Saarland einen Auftraggeber, der unter der Überschrift *Web-based Publishing* einen Anforderungskatalog bereitstellte, das Projekt selbst aber nicht weiter begleitete. Das Institut der Uni-Saarland startete 1998 das europäische Pilot Projekt ITES (Information Technologies in European Sport and Sport Science), das sich aus vier Teilen zusammensetzt, die sich mit jeweils unterschiedliche Zielen internetbasierte virtuelle Kommunikationsmöglichkeiten zunutze machen wollen. Das Ziel der Sport-DL war es, im Internet ein Angebot zu schaffen, das die Möglichkeit bietet, sportwissenschaftliche Fachbeiträge aus dem Bereich Motor Control and Learning zu publizieren sowie umfassende Recherchemöglichkeiten zur Verfügung stellt.

5.1.3 Aufbau

Die Anforderungen und Zielsetzungen des Projektes sind im nächsten Abschnitt genauer präzisiert. Die Vorgehensweise ist im Abschnitt zur Analyse in Form eines Meilensteinplans beschrieben, wobei das hier zugrunde liegende Vorgehensmodell der Rational Unified Process ist und der Schwerpunkt der Analyse auf den Use Cases liegt.
Im Abschnitt Design werden die Benutzerschnittstellen sowie das Datenmodell des Systems vorgestellt.
Da für das Projekt nur ein begrenzter Zeitrahmen zur Verfügung stand, wird auch auf einige externe Werkzeuge zurückgegriffen, die in einem eigenen Abschnitt zusammen mit der eingesetzten Entwicklungsumgebung und dem Datenbankmanagementsystemen näher beschrieben sind.
Auf die Implementationsaspekte, Systemeigenheiten und Systemtests geht der vorletzte Abschnitt genauer ein. Insbesondere werden dort die spezifischen Probleme, die bei der Implementation der Komponenten auftraten, eingehend erläutert.
Das Benutzerhandbuch beschreibt abschließend noch einmal die Grundfunktionalität des Systems für den späteren Endanwender.

5.2 Zusammenfassung der Anforderungsdefinition

Ziel des Subprojektes ist es, eine Sammlung von Dokumenten zu schaffen, die über das WWW abrufbar ist. Das Kernstück ist hierbei das sogenannte e-Journal, welches die Dokumente enthält und Suchfunktionen anbietet.

5.2.1 Die Dokumenttypen des Informationssystems

Das e-Journal verfügt über verschiedene Arten von Dokumenten. Zum einen sind die Volltextdokumente zu nennen, die den Hauptteil des Informationssystems ausmachen. Volltexte sind in diesem Sinne Artikel oder wissenschaftliche Ausarbeitungen, die Eigentum der jeweiligen Autoren sind. Es wird empfohlen, die Dokumente in englischer Sprache zu verfassen. Weiterhin können die Dokumente über mehrere anderssprachige Zusammenfassungen verfügen. Außerdem werden die Artikel einem Reviewverfahren unterzogen, um fehlerhafte oder nicht erwünschte Dokumente aussortieren zu können. Der nächste Dokumenttyp sind die eben schon erwähnten Abstracts oder Zusammenfassungen. Im Rahmen des e-Journals soll es möglich sein, zu bereits vorhandenen Volltexten oder zu anderweitig publizierten Volltexten Abstracts in das System einzustellen. Es kann zu einem Volltext mehrere, anderssprachige Abstracts geben, allerdings sollte eine davon in Englisch verfasst sein. Außerdem sollte die Länge eines Abstracts eine Din-A4-Seite nicht überschreiten. Forschungsnotizen sind eine weitere Klasse von Dokumenttypen. Hierbei handelt es sich um aktuelle Forschungsergebnisse, -fragen und -probleme. Forschungsnotizen sollten eine maximale Länge von 3 Din-A4-Seiten haben und unterliegen keinem Reviewverfahren. Alle weiteren Dokumente, die für das e-Journal interessant und relevant sind, aber nicht klassifiziert werden können, werden zunächst als weitere Dokumente bezeichnet. Bei einer eventuellen Erweiterung der Struktur des e-Journals, kann über eine Eingliederung dieser Dokumente in einen anderen Themenbereich nachgedacht werden.

5.2.2 Einstellungsverfahren

Alle Dokumente müssen im pdf-Format oder Word-Format eingereicht werden. Dies soll zunächst per Email als Attachment geschehen. Nach dem Reviewen des Dokuments wird es dann im pdf-Format in das System eingestellt. Es bleibt abzuklären, ob es Möglichkeiten gibt, das Einstellungsverfahren zu automatisieren und Metadaten aus dem Dokument zu filtern, die für das Suchen im e-Journal verwendet werden könnten. Weiterhin soll jeder im System angemeldete Benutzer Dokumente einstellen können. Dadurch soll das Angebot des e-Journals in kurzer Zeit einen akzeptablen Umfang erreichen.

5.2.3 Funktionen im Informationssystem

Jeder Nutzer des Systems erhält kostenlos eine Zugangsberechtigung. Nachdem er sich einen Nutzernamen ausgesucht und weitere freiwillige Informationen zu seiner Person gemacht hat, wird ihm das initiale Passwort per Email zugesandt. Mit diesem Passwort hat er dann Zugriff auf die verschiedenen Dienste des e-Journals. Dazu gehört beispielsweise das Anzeigen von Dokumenten. Bei Volltexten kann der Nutzer das Dokument über seinen Browser herunterladen oder sich die dazugehörige Literaturliste anschauen. Weiterhin kann er eine Seite aufrufen, die

Links zu themenverwandten Dokumenten enthält. Alle anderen Dokumenttypen stehen sowohl zum Download als auch zur Anzeige bereit. Die Suchfunktionen des e-Journals umfassen nicht nur die übliche Suche nach Autoren und Titeln, sondern auch die Volltextsuche innerhalb der Dokumente. Außerdem bietet das Informationssystem noch Mailinglisten an, die die Nutzer per Email über neu eingestellte Dokumente und sonstige wichtige Ereignisse informiert. Die Kommunikation mit anderen Nutzern findet schließlich mit Hilfe von Diskussionsforen statt. Hier wird zwischen dokumentbezogenen Foren, in denen über bestimmte Dokumente diskutiert werden kann, und allgemeinen Foren unterschieden.

5.2.4 Weitere Datenbestände

Um auch auf Dokumente zugreifen zu können, die anderweitig publiziert worden sind, soll eine Titeldatenbank eingerichtet werden.Innerhalb dieser Datenbank kann nach beliebigen Begriffen gesucht werden. Wobei Dokumente die im e-Journal vorhanden sind direkt heruntergeladen werden können. Ziel ist es schon bestehende Datenbanken mit Volltexten in das System zu integrieren, um das Angebot an Dokumenten zu vergrößern. Weiterhin kann sich jeder registrierte Nutzer in eine Projektseite eintragen. Auf diesen Projektseiten werden lediglich Links zu anderen, externen Seiten verwaltet. Die Aktualität der Links soll von den Nutzern überwacht werden.

5.2.5 Evaluation

Um eine Analyse des Nutzerklientels und dessen Nutzungsverhaltens zu erstellen, sollen die Zugriffshäufigkeiten auf die unterschiedlichen Systembereiche ausgewertet werden. Außerdem sollen im Januar 2000 alle angemeldeten Nutzer eine Email mit einer URL erhalten, die auf einen interaktiven Fragebogen verweist. Alle Nutzer die sich dann neu am System anmelden, müssen zunächst diesen Fragebogen ausfüllen, um Zugang zum System zu bekommen.

5.3 Analyse

Dieses Dokument beschreibt die Ergebnisse der Analyse der zu entwickelnden Sport-DL. Dabei wird zunächst der Meilensteinplan für den Entwicklungszeitraum mit seinen Phasen festgelegt. Anhand der Entwürfe der Benutzerschnittstellen wird die Funktionalität der Sport-DL beschrieben. Dabei werden zwei Sichten unterschieden. Den größeren Teil nimmt die Nutzersicht ein, zu der das Abrufen der in der Sport-DL zur Verfügung gestellten Dokumente und anderen Leistungen wie Mailliste und Newsgroups gehören sowie die Schnittstellen zum Einstellen von neuen Dokumenten in die Digitale Bibliothek. Zur Sicht von Reviewern wird beschrieben, auf welche Art und Weise ein Reviewer Zugang zu den neu eingestellten Dokumenten erhalten kann.

5.3.1 Entwicklungsphasen mit Meilensteinen

Start der Entwicklung ist der 13. Dezember 1999.

1. Bis 20. Dezember 1999: Entwurf der Benutzerschnittstellen

- Leser und Einstellersicht
- Reviewersicht

2. Bis 10. Januar 2000: Entwurf des Datenmodells
3. Bis 17. Januar 2000: Evaluation von Werkzeugen
   - Konvertierungstools (Word nach PDF)
   - Volltextdatenbanken, Indexierungstools
   - Chat-Tools
   - News Foren
   - Tools für Maillisten
4. Bis 31. Januar 2000: Implementation des Datenmodells und der Schnittstellen
5. Bis 22. Fabruar 2000: Implementation der Benutzerschnittstellen
   - Leser und Einstellersicht
   - Reviewersicht (wird aus Zeitgründen nicht durchgeführt)
6. Bis 29. Februar 2000: Test des Gesamtsystems und Verbesserungen

5.3.2 Anwendungfälle

Die Akteure des Systems sind:

Administrator Der Administrator sorgt für die Verfügbarkeit des Systems mit allen seinen Komponenten.

Reviewer Der Reviewer prüft neu eingestellte Dokumente und gibt diese für die Nutzer frei.

Nutzer Die Nutzer sind sowohl die Leser, als auch Einsteller von Dokumenten.

Datenbanken Die Dokumentdatenbank zur Speicherung der Metadaten und sonstiger Dokumentinformationen sowie die Nutzerdatenbank mit Informationen über die registrierten Benutzer.

Newsgroupsystem Das Newsgroupsystem verwaltet die zu den einzelnen Themen eingerichteten Gruppen.

Mailinglistenverwalter Der Mailinglistenverwalter verteilt die e-mails zu den neu eingestellten Dokumenten an die eingetragenen Nutzer.

Für die im Folgenden beschriebenen Use-Cases gilt immer die Vorbedingung, dass sich der Nutzer erfolgreich beim System registriert hat.

5.3.2.1 Registration beim System

Dieser Use-Case beschreibt die Registration eines Nutzers bei der Sport-DL. Der Nutzer muß einige Daten bei der Sport-DL angeben um dann in Form eines Logins und eines Passworts eine Zugangsberechtigung zum System zu erhalten.

Akteure: Nutzer, Nutzerdatenbank

1. Der Nutzer hat die Startseite der Sport-DL angewählt, klickt auf den Link *Registration* und bekommt das Formular zur Registration auf seinem Browser dargestellt.
2. Der Nutzer gibt seinen Benutzernamen und seine E-Mailadresse in das Formular ein und schickt die Daten über einen Submitbutton an das System. Optional können auch weitere Nutzerinformationen angegeben werden.
3. Das System prüft die Daten und schickt an die angegebene E-Mailadresse eine Mail mit dem Benutzernamen und dem Passwort des Nutzers.

Ausnahmen / Varianten:
Ergibt die Prüfung der Daten in 3, dass der Nutzer in (b) einen Namen angegeben hat, der schon von einer anderen Person benutzt wird, vergibt das System ein modifiziertes Login.

5.3.2.2 Anmeldung beim System

Dieser Use-Case beschreibt den Anmeldevorgang eines Nutzers der Sport-DL. Durch die Anmeldung wird dem Nutzer die Zugangsberechtigung zu den Seiten der Sport-DL erteilt.

Akteure: Nutzer, Nutzerdatenbank

1. Der Nutzer hat die Startseite der Sport-DL angewählt, klickt auf den Link *Login* und bekommt das Loginformular auf seinem Browser dargestellt.
2. Der Nutzer gibt seinen Benutzernamen und sein Paßwort in das Formular ein und schickt die Daten über einen Submitbutton an das System.
3. Das System verifiziert die Nutzerdaten anhand der Nutzerdatenbank und ruft die Wilkommenseite der Sport-DL auf.

Ausnahmen / Varianten:
Stellt das System bei Schritt 3 einen falschen Benutzernamen oder Passwort fest, erscheint eine entsprechende Mitteilung auf der Loginseite und der Benutzer beginnt den Use-Case bei Schritt 1.

Für die im Folgenden beschriebenen Use-Cases gilt immer die Vorbedingung, dass sich der Nutzer erfolgreich beim System, wie im Use-Case *Anmelden beim System* beschrieben, angemeldet hat.

5.3.2.3 Volltexte anzeigen

Dieser Use-Case beschreibt die Navigation eines Nutzers der Sport-DL, um ein Volltextdokument aus der Datenbank abrufen und online lesen zu können.

Akteure: Nutzer, Dokumentendatenbank

1. Der Nutzer wählt die Funktion *Im e-Journal blättern*, wodurch die entsprechende Seite im Browser angezeigt wird.
2. Das System fragt die Dokumentendatenbank nach allen Volltexten ab und generiert aus dem Ergebnis eine Html-Seite, die dem Nutzer angezeigt wird.
3. Der Nutzer wählt nach dem Sichten der Seite zu einem Volltext die Funktion Anzeigen durch Klick auf einen der Titel aus.
4. Das System ruft den Volltext aus der Dokumentendatenbank ab und sendet ihn an den Browser des Nutzers.
5. Verfügt der Nutzer über ein Plugin für seinen Browser, wird dies vom Browser geladen und der Volltext wird angezeigt.

Ausnahmen / Varianten:
Besitzt der Nutzer bei 5. kein Plugin so ist der weitere Ablauf hier vom Browser abhängig.

5.3.2.4 Abstract herunterladen

Dieser Use-Case beschreibt die Navigation eines Nutzers der Sport-DL, um eine Abstract (Zusammenfassung) aus der Datenbank auf seinem PC abspeichern zu können.

Akteure: Nutzer, Dokumentendatenbank

1. Der Nutzer wählt die Funktion *In den Abstracts blättern*, wodurch die Entsprechende Seite im Browser angezeigt wird.
2. Das System fragt die Dokumentendatenbank nach Abstracts ab und generiert aus dem Ergebnis eine Html-Seite, die dem Nutzer angezeigt wird.
3. Der Nutzer wählt nach dem Sichten der Seite zu einem Abstrakt die Funktion Download des Dokuments durch Klick auf einen Button aus.
4. Das System ruft das Abstract aus der Dokumentendatenbank ab und sendet es an den Browser des Nutzers.
5. Die Möglichkeiten des Nutzers ein Ziel für den Download anzugeben oder den Download abzubrechen werden vom jeweiligen Browser zur Verfügung gestellt und liegen somit außerhalb des Systems. Der Use-Case endet, wenn die Datei vollständig auf dem lokalen Rechner des Nutzers zur Verfügung steht.

Ausnahmen / Varianten:
keine

5.3.2.5 Forschungsnotizen abrufen

Dieser Use-Case beschreibt die Navigation eines Nutzers der Sport-DL, um eine Forschungsnotiz aus der Datenbank abrufen und online lesen zu können.

Akteure: Nutzer, Dokumentendatenbank

1. Der Nutzer wählt die Funktion *Forschungsnotizen anzeigen / blättern*, wodurch die Entsprechende Seite im Browser angezeigt wird.
2. Das System fragt die Dokumentendatenbank nach Forschungsnotizen ab und generiert aus dem Ergebnis eine Html-Seite die dem Nutzer angezeigt wird.
3. Der Nutzer wählt nach dem Sichten der Seite zu einer Forschungsnotiz die Funktion Anzeigen durch Klick auf einen Button aus.
4. Das System ruft den Forschungsnotiz aus der Dokumentendatenbank ab und sendet ihn an den Browser des Nutzers.
5. Verfügt der Nutzer über ein Plugin für seinen Browser wird dies vom Browser geladen und der Volltext wird angezeigt.

Ausnahmen / Varianten:
Besitzt der Nutzer bei 5. kein Plugin so ist der weitere Ablauf hier vom Browser abhängig. Auf einer Hilfeseite der Sport-DL kann sich der Nutzer aber informieren, welches Plugin benötigt wird und wo es zum Download zur Verfügung steht.

5.3.2.6 Suche

Dieser Use-Case beschreibt die Navigation eines Nutzers der Sport-DL, um Dokumente (Volltext, Abstracts, Forschungsnotizen etc.) in der Datenbank über die einfache Suche zu suchen.

Akteure: Nutzer, Dokumentendatenbank

1. Der Nutzer wählt die *Funktion Suche / Navigation*, wodurch die Entsprechende Seite im Browser angezeigt wird.
2. Der Nutzer trägt im Formularfeld ein Stichwort ein zu dem er Texte finden möchte und wählt in dem Auswahlfeld, ob das angegebene Stichwort im Titel de Dokuments oder im Namen des Autors vorkommen soll.
3. Der Nutzer schickt durch Klicken des Suchen Buttons die Suchanfrage an das System.
4. Das System fragt die Dokumentendatenbank nach Dokumenten zu dem Suchbegriff ab und generiert aus dem Ergebnis eine Html-Seite mit den gefundenen Dokumenten, die dem Nutzer angezeigt wird.

Ausnahmen / Varianten:
keine

5.3.2.7 Dokumente einstellen

Dokumente, also Volltexte, Abstracts und Forschungsnotizen können von beliebigen Personen in die Sport-DL eingespeist werden.
Unter Dokumente einstellen ist das Einstellen von Volltexten, Forschungsnotizen und Abstracts zu Volltexten, die bereits in das e-journal eingestellt wurden, möglich. Da dies getrennt voneinander durchgeführt werden soll, werden für die jeweiligen Dokumentarten Varianten des Use-Cases angegeben.

Akteure: Nutzer, Dokumentendatenbank

1. Der Einsteller klickt auf den Link Dokumente einstellen und bekommt ein eine Html-Seite dargestellt, von der aus er zu den drei Formularen für die unterschiedlichen Dokumentarten gelangt.

2. Unterscheidung von drei Fällen:
   i: Der Einsteller wählt Volltextdokument einstellen. Weiter mit (c).
   ii: Der Einsteller wählt Forschungsnotiz einstellen. Weiter mit (c).
   iii: Der Einsteller gibt den Titel des Volltextes ein, zu dem er ein Abstract einstellen will und klickt auf den Suchen Button. Das System sucht in der Dokumentdatenbank die der Suchanfrage entsprechenden Volltextdokumente und schickt eine Html-Seite mit den Ergebnissen an den Nutzer. Der Nutzer wählt Abstract einstellen zu dem gewünschten Dokument. Weiter mit (c).

3. Der Nutzer bekommt ein Formular zur Angabe der Dokumentinformationen und der benötigten Datei(en) im Browser dargestellt.

4. Der Nutzer gibt die Metadaten zu dem Dokument ein.

5. Der Nutzer gibt den (die) Pfad(e) der Datei(en) des Dokuments ein.
   Für Volltexte müssen die Volltextdatei sowie die dazugehörige Abstractdatei angegeben werden. Das Einstellen einer separaten Literaturliste ist optional.
   Beim Einstellen von Forschungsnotizen und Abstracts zu bereits vorher eingespielten Volltexten muß jeweils nur eine Datei angegeben werden.

6. Der Nutzer klickt auf den Button abschicken.

Ausnahmen / Varianten:
keine

5.3.2.8 In Mailing-Liste eintragen

Dieser Use-Case beschreibt, wie sich ein Nutzer in eine der Mailinglisten eintragen kann, um per e-mail Informationen zu neu eingestellten Dokumenten zu erhalten.

Akteure: Nutzer, Nutzerdatenbank

1. Der Nutzer klickt auf den Link Kommunikation und bekommt eine Seite dargestellt, auf der verschiedene Mailinglisten zur Auswahl stehen.
2. Nun kann der Nutzer entscheiden, zu welchen Dokumenten er Informationen erhalten will, indem er die entsprechenden Checkboxen aktiviert.
3. Der Nutzer klickt auf die Schaltfläche Anmelden.
4. Das System nimmt in der Nutzerdatenbank Eintragungen vor, die den Nutzer für die jeweiligen Mailinglisten als ein- bzw. ausgetragen markieren.

Ausnahmen / Varianten:
keine

5.3.2.9 Von Mailing-Liste abmelden

Dieser Use-Case beschreibt, wie sich ein Nutzer aus einer der Mailinglisten austragen kann.

Akteure: Nutzer, Nutzerdatenbank

1. Der Nutzer klickt auf den Link Kommunikation und bekommt eine Seite dargestellt, auf der verschiedene Mailinglisten zur Auswahl stehen.
2. Nun kann der Nutzer entscheiden, zu welchen Dokumenten er keine Informationen mehr erhalten will, indem er die entsprechenden Checkboxen deaktiviert.
3. Der Nutzer klickt auf die Schaltfläche Anmelden.
4. Das System nimmt in der Nutzerdatenbank Eintragungen vor, die den Nutzer für die jeweiligen Mailinglisten als ein- bzw. ausgetragen markieren.

Ausnahmen / Varianten:
keine

5.3.2.10 Nachricht in den Newsgroups lesen

Dieser Use-Case beschreibt, wie der Nutzer zu der Angeschlossenen Newsgroup gelangt und Nachrichten darin abrufen kann.

Akteure: Nutzer, Newsgroupsystem

1. Der Nutzer klickt auf den Link Kommunikation und bekommt eine Seite dargestellt mit den vorhandenen Newsgroups dargestellt.
2. Der Nutzer klickt auf einen Link zu einer Newsgroup und gelangt damit zum Newsgroupsystem mit den Nachrichten.

3. Nun klickt der Nutzer auf die Nachricht, die er lesen will und bekommt diese dann in seinem Browser angezeigt.

Ausnahmen / Varianten:
keine

5.3.2.11 Auf Nachricht in den Newsgroups antworten

Dieser Use-Case beschreibt, wie der Nutzer zu der Angeschlossenen Newsgroup gelangt und Nachrichten darin beantworten kann.

Akteure: Nutzer

1. Der Nutzer klickt auf den Link Kommunikation und bekommt eine Seite dargestellt mit den vorhandenen Newsgroups dargestellt.
2. Der Nutzer klickt auf einen Link zu einer Newsgroup und gelangt damit zum Newsgroupsystem mit den Nachrichten.
3. Nun klickt der Nutzer auf die Nachricht, die er beantworten will.
4. Dann klickt der Nutzer auf den Link Antworten.
5. Der Nutzer trägt seinen Namen, optional seine E-Mail-Adresse und das Thema seiner Antwort in die dafür vorgesehenen Textfelder ein.
6. Dann tippt er den Antworttext in die Textbox und klickt auf die Schaltfläche Abschicken. Damit ist die Nachricht in die Newsgroup eingetragen.

Ausnahmen / Varianten:
keine

5.3.3 Use Case Diagramm mit den Anwendungsfällen des Nutzers

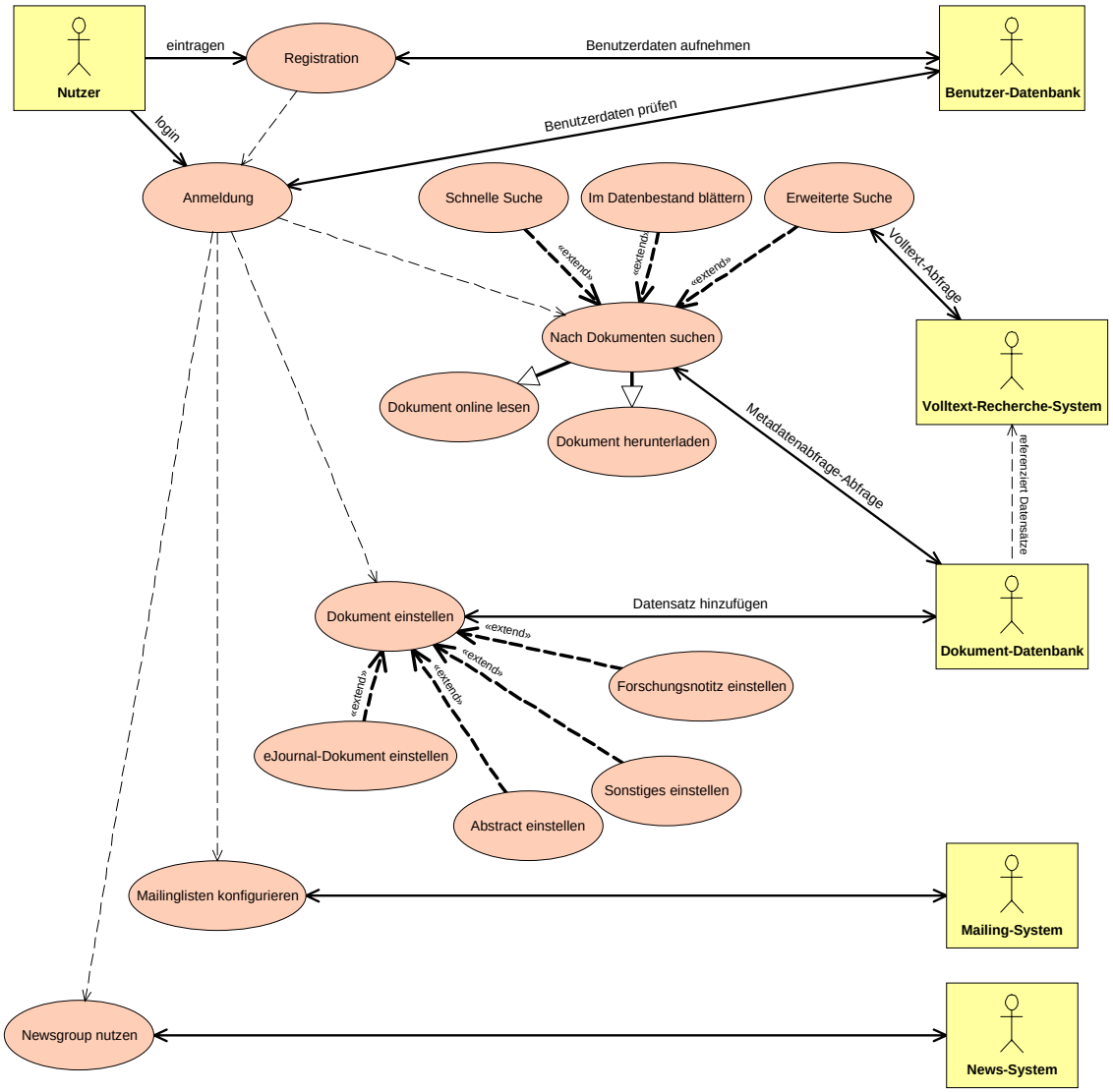

Abbildung 5.1: Use Case Diagramm mit den Anwendungsfällen des Nutzers

5.4 Design

5.4.1 Das Datenmodell zur Sport-DL

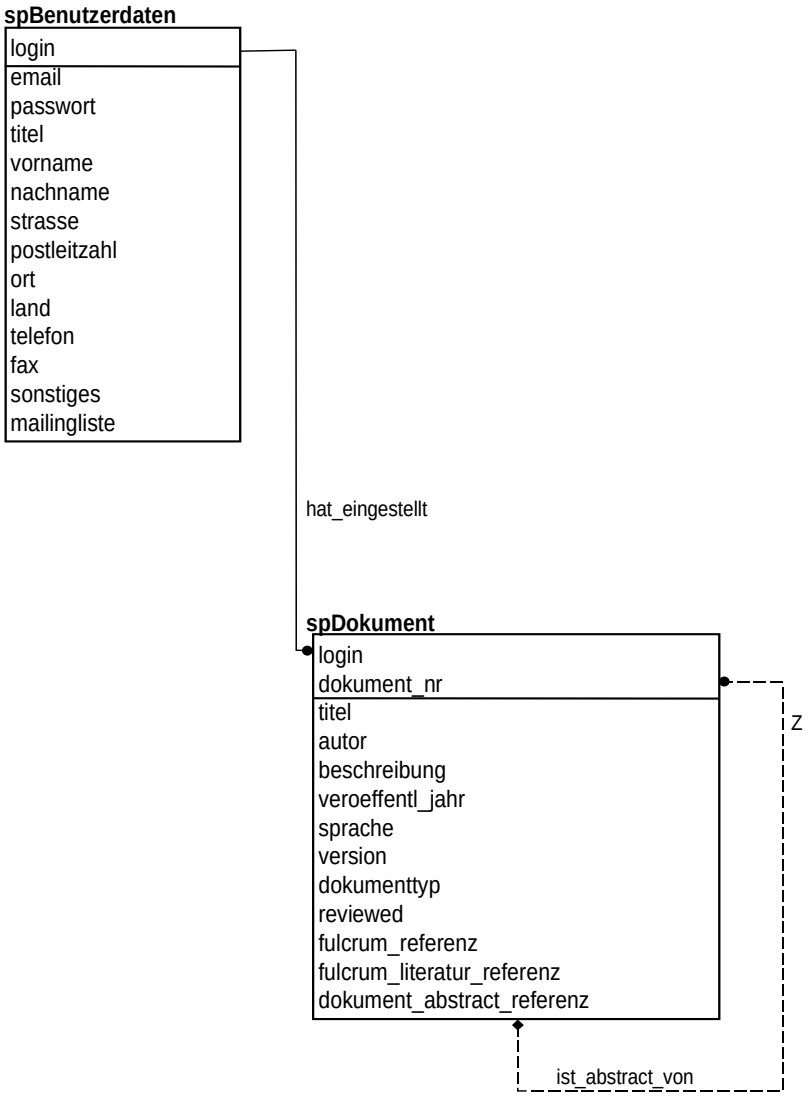

Abbildung 5.2: Das Datenmodell zur Sport-DL

5.4.2 Definition der Benutzerschnittstellen

5.4.2.1 Die Sport-DL Startseite

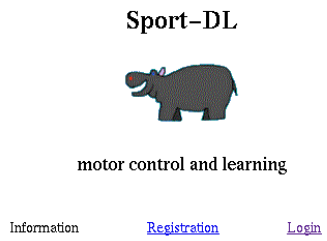

Abbildung 5.3: Die Startseite der Sport-DL

Der Benutzer hat die Möglichkeit, sich über die SportDL zu informieren, sich einzuloggen, sofern er schon einen Account bei der SportDL hat oder sich bei der SportDL zu registrieren.

5.4.2.2 Die Registration eines neuen Nutzers

Abbildung 5.4: Die Registration

Bei der Registration wird der Benutzer dazu aufgefordert, einige benutzerspezifische Daten anzugeben. Sein Name und seine E-Mail-Adresse sind dabei besonders wichtig, da der Benutzer beim Einloggen in der SportDL mit Namen vom Programm angesprochen wird und die SportDL Mailinglisten und Newsseiten zur Verfügung stellt, wo die E-Mail-Adresse benötigt wird. Dem Benutzer wird ausserdem nach der Anmeldung bei der SportDL sein Passwort per E-Mail zugestellt.

5.4.2.3 Die Anmeldung eines Nutzers über sein Login

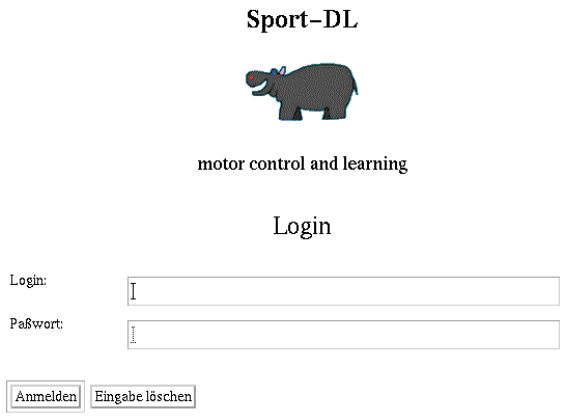

Abbildung 5.5: Login

Hat der Benutzer sich schon in der SportDL registriert, so kommt er über Login auf diese Seite wo er seinen Benutzernamen und sein Passwort angeben muß.

5.4.2.4 Die Startseite mit den Suchefunktionen

Abbildung 5.6: Startseite nach dem Einloggen mit Möglichkeiten der Suche

Hat der Benutzer sich erfolgreich eingelogged, dann wird ihm diese Seite dargestellt, die sofort die Möglichkeit zum Suchen nach Dokumenten in der SportDL zur Verfügung stellt. Dabei

kann er entweder einen *Suchbegriff* eingeben oder die Archive des *eJournals*, der *Abstracts*, der *Forschungsnotizen* oder im *Sonstiges-Bereich* selbst nach Dokumenten suchen. Sollte er gezielt ein Dokument suchen wollen, zu dem er noch weitere Angaben hat, so kann er über die *Erweiterte Suche* (siehe Abb. 5.7) fortfahren. Alle Ergebnisse einer erfolgreichen Suchanfrage werden unabhängig von der gewählten Suchfunktion, immer in der gleichen Form bzw. Layout dargestellt (siehe Abb. 5.8).

Im Kopfbereich dieser Seite (Abb. 5.6) befindet sich eine Navigationsleiste, die sich auf jedem weiteren Bildschirm oben wiederholt. Hier hat der Benutzer die Möglichkeit, egal auf welcher Seite er sich gerade befindet, gezielt eine Funktion aus dem Angebot der SportDL anzuwählen.

5.4.2.5 Die erweiterte Suche mit Volltextsuche

Abbildung 5.7: Erweiterte Suche

Hier hat der Benutzer mehrere Möglichkeiten, die Suche einzuschränken.

5.4.2.6 Die Darstellung der Suchergebnisse

Abbildung 5.8: Beispiel zu den Suchergebnissen

5.4.2.7 Neue Dokumente einspielen - festlegen der Dokumentart

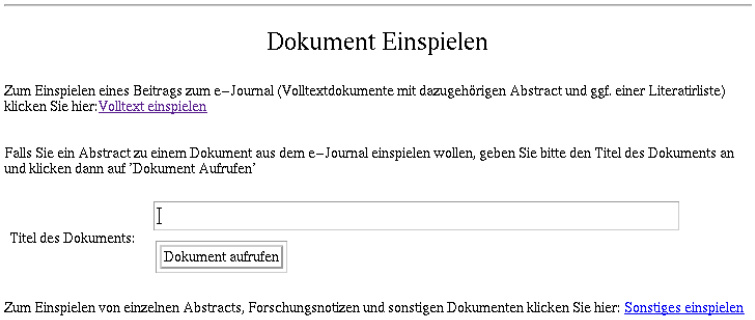
Dokument Einspielen
Zum Einspielen eines Beitrags zum e–Journal (Volltextdokumente mit dazugehörigen Abstract und ggf. einer Literatirliste) klicken Sie hier: Volltext einspielen
Falls Sie ein Abstract zu einem Dokument aus dem e–Journal einspielen wollen, geben Sie bitte den Titel des Dokuments an und klicken dann auf 'Dokument Aufrufen'
Titel des Dokuments:
Dokument aufrufen
Zum Einspielen von einzelnen Abstracts, Forschungsnotizen und sonstigen Dokumenten klicken Sie hier: Sonstiges einspielen

Abbildung 5.9: Dokumente einspielen

Um neue Dokumente einzuspielen, muss sich der Benutzer zwischen drei Wegen entscheiden, abhängig vom Typ des Dokuments, dass er einspielen möchte. Die Links auf dieser Seite führen zu den jeweiligen Formularen, über die der Benutzer die notwendigen Informationen zu seinem Dokument zur Verfügung stellt.

Falls der Benutzer ein neues Abstract zu einem Dokument aus dem eJournal einspielen möchte, so kann in dem Textfeld den Titel, oder Teile des Titels, des eJournal-Dokuments eingeben, und sein Abstract so gezielt einem Dokument zuordnen (siehe Abb. 5.11).

Für das Einspielen von Volltexten sind nochmehrere Angaben nötig, weswegen er dafür

über den link Volltext Einspielen gehen muss (siehe Abb. 5.10).

Für alle sonstigen Dokumenttypen, wie Forschungsnotizen, einzelne Abstracts zu externen Dokumenten oder andere Berichte, wird der untere Link benutzt (siehe Abb. 5.13).

5.4.2.8 Volltexte für das eJournal einspielen

Abbildung 5.10: Volltext einspielen

Für das Einspielen eines Volltextes werden mehrere Informationen vom Benutzer benötigt, u.a. eine Literaturliste und ein Abstract. Diese beieden Dokumente müssen als Word- oder PDF-Dateien vorliegen.

5.4.2.9 Einspielen eines zusätzlichen Abstracts zu einem vorliegenden Volltext

Volltexte
Zu den folgenden Volltexten können Sie Abstracts einstellen:
1. : Volltext
Rockmann, U. & Thielke, St.,
Zum Einsatz von hypermedialen Lernprogrammen und internetfähigen Applikationen
1999
Abstract einspielen

Abbildung 5.11: Abstract zu einem speziellen Volltext nachträglich einspielen

Sofern man einen Abstract zu einem bestehenden Volltext einspielen möchte, wird eine Liste von verfügbaren Volltexten präsentiert um zu gewährleisten, dass der Abstract auch zum richtigen Volltext eingespielt wird.

5.4.2.10 Ein zusätzliches Abstract einspielen

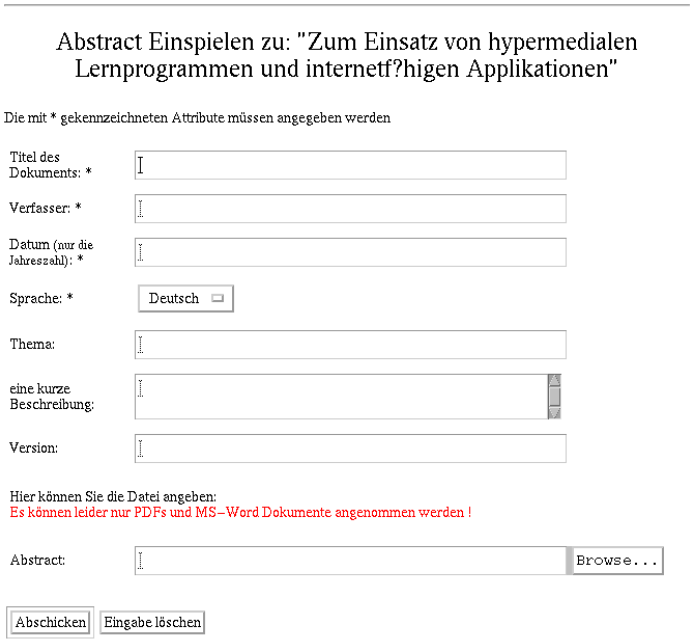

Abbildung 5.12: Eingabe der Daten zu einem zusätzlichen Abstract

Hat man nun in der vorherigen Liste den gewünschten Volltext gefunden, zu dem der Abstract eingespielt werden soll, so kommt man auf diese Seite, wo wieder Informationen bzgl. des Abstracts und die Datei des Abstracts angegeben werden müssen.

5.4.2.11 Sonstige Dokumente einspielen

Abbildung 5.13: Einspielen von sonstigen Dokumenten

Wie auch beim Abstract werden hier zu sonstigen Dokumenten Informationen zu dem einzuspielenden Dokument und das Dokument selber angegeben.

5.4.2.12 Die Kommunikationsangebote der Sport-DL

Abbildung 5.14: Kommunikation in der Sport-DL

Hier hat man die Möglichkeit per Link auf die Seiten des Hypernews-Systems zu gelangen oder sich bei den Mailinglisten der SportDL an- oder abzumelden. Bei den Mailinglisten wird per Haken dargestellt, wo man schon angemeldet ist.

5.4.2.13 Verändern der persönlichen Benutzerdaten

Abbildung 5.15: Konfiguration der Benutzereinstellungen

Sollten sich die Benutzerdaten geändert haben, so kann der Benutzer auf dieser Seite die gewünschten Änderungen vornehmen. Insbesondere soll diese Seite benutzt werden, um Passwörter ändern zu können.

5.5 Evaluation

5.5.1 Komponenten, die nachher verwendet wurden

Betriebssystem : Solaris
Web-Server : Apache
Sprache : Java mit Servlets
Newsserver : HyperNews
Standard Datenbank : Oracle 8
Volltext Datenbank : Fulcrum
Text Editor : Emacs
Mailingliste : MajorDomo
Browser : Netscape
Rechner : Sun Sparc

Mail Tool : JavaMail

5.5.2 Komponenten, die vorausgesetzt werden mußten

Die zur Verfügung gestellten Rechner waren SUN Sparc-Stations. Da es sich nur um die Erstellung eines Prototypen bei dem Projekt handelt und kein weiterer Einsatz innerhalb des OFFIS geplant ist, war die Anschaffung eines neuen Rechners für die Installation der Software nicht notwendig.
Das Betriebssystem Solaris mußte vorausgesetzt werden, da kein Rechner zur Verfügung war, auf dem ein anderes Betriebssystem aufgespielt war, bzw auf den ein anderes Betriebssystem hätte installiert werden können.
Bei der Nutzung der Rechner und des Betriebssystems traten keine Probleme oder Fehler auf.

5.5.3 Komponenten, die keiner größeren Evaluation unterworfen wurden

Es wurde entschieden, den Web-Server Apache zu nutzen, ohne daß dieses Programm von der Gruppe vorher näher untersucht worden ist. Die Gründe dafür waren, daß einige aus der Gruppe schon vorher gute Erfahrungen mit Apache gemacht hatten, Apache innerhalb des OFFIS bei vielen Mitarbeitern zufriedenstellend genutzt wird und die Gruppenbetreuerin Cornelia Haber sich bereit erklärt hatte, die Installation des Apache vorzunehmen und auch für die Wartung zuständig war.
Wir sind mit dem Laufzeitverhalten des Apache sehr zufrieden, jedoch stellte sich das Testen der Java-Servlets am Apache als unangenehm dar, weil es ausser einem Internal Server Error keine Fehlermeldung vom Apache gab, die Fehler innerhalb der Servlets näher beschrieben. Ausserdem mußte Apache jedesmal neu gestartet werden, wenn ein Servlet neu eingespielt oder verändert worden war. Zu beachten ist, daß Apache unter die GPL fällt und damit kostenlos ist.
Als standard Datenbank wurde Oracle 8 benutzt. Diese Datenbank wurde uns vorgegeben, da sie im OFFIS schon längere Zeit erfolgreich eingesetzt wird und keine Zeit vorhanden war, mehrere Alternativen auszuprobieren, was auch bedeutet hätte, daß man sich mit deren Schnittstellen und der Ansteuerung dieser Datenbanken (z.B. MySQL) hätte auseinandersetzen müssen. Ausserdem wurden uns Routinen zur Verfügung gestellt, mit denen wir die Datenbank von Java-Servlets aus ansteuren konnten (JDBC-ODBC).
Beim Einsatz von Oarcle traten Probleme im Laufenden Betrieb bei dem Löschen von Tabellen auf.
Die gleichen Voraussetzungen trafen auch auf die Datenbank Fulcrum zu, die zur Indexierung von Volltexten genutzt wurde. Da wir nur eine sehr alte Version von Fulcrum zur Verfügung hatten, traten Probleme beim Indexieren von neueren PDF-Dateien auf. Diese Probleme sollen aber mit einer neueren Version von Fulcrum gelöst werden.
Aufgrund der Konzeption der digitalen Datenbank hatten wir uns sehr früh für die Programmiersprache Java und die Erstellung von Java-Servlets entschieden, da hiermit die Realisierung einer serverseitigen Anbindung der Datenbanken möglich ist. Aus diesem Grund haben wir uns nicht näher mit der Evaluierung von Java beschäftigt.
Zur Realisierung einer Mailingliste wurde MajorDomo verwendet, da dieses Programm schon auf den Rechnern installiert war und wir nur bei dem Administrator dieses Tools das Einrichten neuer Listen anfordern mußten. Die Nutzung dieser Listen verlief problemlos.
.

5.5.4 Komponenten, die einer vollständigen Evaluierung unterworfen wurden

5.5.4.1 HyperNews

Es wurde ein News-Server benötigt, der als Diskussionsforum dienen sollte.Es wurde überlegt, ob dafür die Mailinglisten MailMan oder MajorDomo benutzt werden könnten. Diese Programme hätte man über Perl-Skripte in die Lage versetzen können, daß eingehende E-Mails in Form von HTML-Seiten archiviert werden können. Da sich bei uns niemand mit der Programmierung von Perl-Skripten auskennt, wurde diesen Möglichkeiten aufgrund von Zeitmangel verworfen.
Von dem Projektgruppenbetreuer Dietrich Boles kam dann der Vorschlag, HyperNews einzusetzen. HyperNews ist eine News-Server-Software, die es ermöglicht, News am Browser einzugeben oder zu lesen.HyperNews besteht aus Perl-Skripten, die über die CGI-Bin Schnittstelle des Apache gestartet werden.
HyperNews beinhaltet eine Benutzerverwaltung, welche wir allerdings nicht funktionsfähig bekommen haben. Dadurch war es nicht möglich, News zu lesen oder zu schreiben. Aus diesem Grund haben wir uns entschlossen, die Rechteverwaltung für Benutzer über das Administrations-Skript ausser Kraft gesetzt. Alle Perl-Skripte, mit denen die Benutzer konfrontiert werden, sind editiert worden, damit das Arbeiten mit HyperNews vereinfacht wird und die Benutzerverwaltung vor dem Benutzer verborgen bleibt. Es kann zwar jeder Nutzer des Internets News in HyperNews veröffentlichen, was nicht gewünscht war, aber da der Administrator von HyperNews von jeder News-Meldung eine Kopie per E-Mail bekommt, wird so sichergestllt, daß nicht längerfristig Schaden in Form der Verbreitung von diskriminierenden oder illegalen Informationen auf dem News-Server möglich ist.
Aufgrund von Zeitmangel war es nicht möglich, neben HyperNews noch einen weiteren News-Server auszuprobieren. Für den Fall, daß eine überarbeitete Version des Prototyps später genutzt werden soll, wird empfohlen, eine Alternative zu HyperNews zu wählen.

5.5.4.2 JavaMail

JavaMail ist eine Java-Klasse, die es ermöglicht, von einem Java-Programm aus eine E-Mail zu schreiben. Die Konfiguration und Compilation von JavaMail war nicht sehr kompliziert und die Nutzung verlief problemlos.

5.5.5 Komponenten, die unabhängig vom Projekt waren

Um den Prototyp zu schreiben und zu testen wurde mehrere Programme benutzt, die von den Benutzern und Informatikern nach subjektivem Gefallen gewählt wurden. Als Internet-Browser zum Laden und darstellen des Prototyps wurde Netscape benutzt. Die Programme wurden mit dem Emacs geschrieben bzw. editiert. Der Vorteil beim Emacs als Editor ist das farbige Darstellen des Programm-Codes, was uns bei der Fehlersuche sehr hilfreich war.

5.6 Implementierung und Test

5.6.1 Einsatz zusätzlicher Tools und Komponenten

In der Implementationsphase wurden die folgenden externen Komponenten und Pakete eingesetzt, die z.T. schon im Abschnitt Evaluation näher beschrieben wurden.

Apache Webserver

Java 1.2 Entwicklungsumgebung mit den Nicht-Standardpaketen:

- **JavaMailAPI** Schnittstelle zum Versenden von Mails aus Java-Programmen
- **OReilly** Paket zur Verarbeitung von Servlet-Requests und File-Uploads
- **eVerlage** Schnittstelle des Offis-Projekts eVerlage für die Kommunikation mit dem Volltextdatenbankmanagementsystem Fulcrum

Oracle DBMS

Fulcrum DBMS zur Volltextrecherche

Hypernews Newssystem

Majordomo Tool zum Verwalten von Mailinglisten

5.6.2 Implementierungsaspekte

Die funktionalen Anforderungen des Systems, so wie sie im Abschnitt Analyse in form von Use-Cases beschrieben sind, werden von unterschiedlichen Servlets umgesetzt. Jedes Servlet setzt sich aus Gründen der Übersichtlichkeit und besseren Wartbarkeit nur mit einer genau abgegrenzten Funktionalität des Gesamtsystems auseinander.

Im Folgenden werden nur die Servlets näher beschrieben, die eine komplexe Funtionalität bieten. Die Implementierung der einfachen Suche sowie das Blättern in den einzelnen Archiven ist trivialer Natur und bedient sich weit weniger Funktionen wie die erweiterte Suche.

5.6.2.1 Registration bei der SportDL

Servletname: SportDL_Registration

Benutzte Klassen und Pakete: java.sql, java.io, javax.servlet, javax.servlet.http, SportDL_ODBC, SportDL_Sitepool, SportDL_Mail

Grundsätzliche Arbeitsweise: Die für die Registration notwendigen Daten werden aus dem Web-Formular extrahiert. Lediglich das Login und die Email-Adresse müssen eingegeben werden. Die restlichen Angaben sind freiwillig. Allerdings gelten für diesen beiden Daten syntaktische Einschränkungen: Die Email-Adresse muss das at-Zeichen '@' enthalten und das Login muss mindestens 5 Zeichen lang sein. Sind die Eingaben korrekt, so werden die Daten in der Datenbank abgespeichert. Zunächst wird noch überprüft, ob das Login schon vorhanden ist. Existiert das Login schon, dann wird eine fortlaufende Nummer

angehängt, um eine Doppelbelegung zu verhindern. Nun wird eine Email zusammengesetzt, die das Login und das Passwort des neuen Nutzers enthält. Diese Email wird an die angegebene Adresse versandt und der neue Nutzer kann sich mit diesen Daten am System anmelden. Nun wird noch eine HTML-Seite aufgebaut, die einen Link auf die Homepage der SportDL beinhaltet. Wurden jedoch nicht korrekte Eingaben gemacht, so erscheint eine entsprechende Fehlermeldung.

Probleme bei der Implementation: Erwähnenswert ist an dieser Stelle, dass bei der Zusammensetzung der SQL-Statements darauf geachtet werden muss, jedes einzelne Datum in Hochkommata zu setzen. Ansonsten werden die Daten nicht korrekt abgespeichert.

5.6.2.2 Anmeldung bei der SportDL

Servletname: SportDL_Login

Benutzte Klassen und Pakete: java.sql, java.io, javax.servlet, javax.servlet.http, SportDL_ODBC, SportDL_Sitepool

Grundsätzliche Arbeitsweise: Das Login und das Passwort werden aus dem Formular entnommen. Dann wird überprüft, ob dieses Datenpaar in der Oracle-Datenbank vorhanden ist. Ist dies der Fall, so wird die Hauptseite der SportDL aufgebaut, die die Navigation, die einfache Suche und das Dokumentarchiv enthält. Stimmen Passwort und Login jedoch nicht überein, so wird eine Fehlermeldung ausgegeben und der Nutzer erhält keinen Zugang zur SportDL.

5.6.2.3 Herunterladen von Dokumenten

Servletname: SportDL_Dokument

Benutzte Klassen und Pakete: java.sql, java.io, javax.servlet, javax.servlet.http, com.oreilly.servlet, SportDL_ODBC, SportDL_Sitepool

Grundsätzliche Arbeitsweise: Wurde in einer Liste von Dokumenten auf den Button 'Download' geklickt, so wird das entsprechende Dokument mit dem Mime-Type 'application/sportdl-pdf' an den Browser geschickt. Da dieser den Mime-Type nicht erkennt, weil er frei erfunden ist, versucht er die Datei abzuspeichern und fordert den Nutzer auf, ein Verzeichnis und einen Dateinamen auszusuchen. Dann wird das Dokument zum Browser übertragen.

Probleme bei der Implementation: Das Anzeigen von Dokumenten sollte zunächst ähnlich implementiert werden. Es traten jedoch unerklärliche Übertragungsfehler auf. Daraufhin wurde entschieden, das Anzeigen von Dokumenten über Links zu realisieren.

5.6.2.4 Erweiterte Suche in der Datenbank

Servletname: SportDL_ErwSuche

Benutzte Klassen und Pakete: java.sql, java.io. javax.servlet, javax.servlet.http, SportDL_ODBC, SportDL_FCDB, SportDL_Sitepool

Grundsätzliche Arbeitsweise: Grundlage für die Arbeitsweise sind die Daten, die der Benutzer in das Formular für die erweiterte Suche eingetragen hat. Die wichtigsten Felder sind dabei die drei Datenfelder zusammen mit den jeweiligen Auswahlfeldern, die den Typ eines Datenfeldes und die Verknüpfung der Felder bestimmen. Aus den Inhalten der Formularfelder wird in mehreren Schritten ein SQL-Statement zusammengesetzt, welches zunächst noch keine Elemente für eine Volltextsuche enthält, also nur Metadaten wie Titel, Autor, Veröffentlichungszeitpunkt und Dokumenttyp berücksichtigt. Dieses Statement wird als Query bzw. Anfrage an die Oracle-Datenbank (hier die Tabelle spDokument) weitergereicht. Das Ergebnis (bzw. Result-Set) wird anschließend Datensatz für Datensatz ausgewertet. Enthält das Formular ebenfalls einen oder mehrere Einträge für eine Volltextsuche, werden diese auf jeden Datensatz aus dem bisherigen Suchergebnis angewendet. Bei der gegenwärtigen Implementation der Volltextsuche ist das Kriterium *Volltext* also stärker als *Autor* oder *Titel*. Außerdem werden Volltextausdrücke als eigene Gruppe ausgewertet (Beispiel: Bei einer Suchanfrage „Volltextausdruck1 `oder` Autor2 `und` Volltextausdruck3, Veröffentl.-jahr = frei, Einschränkung auf eJournal-Beiträge“ werden zunächst in der Oracle-Tabelle spDokument alle eJournal-Beiträge von Autor2 gesucht und anschließend verglichen ob sowohl Volltextausdruck1 als auch Volltextausdruck3 in diesen Dokumenten enthalten sind.). Für eine detailliertere korrekte Auswertung stand in der Implementationsphase nicht genügend Zeit zur Verfügung.

Bei erfolgreicher Suche wird dem Anwender eine Ergebnisseite präsentiert, die über eine Instanz der Hilfsklasse SportDL_Sitepool aufgebaut wird. Als Ergebnis der Suche wird eine Liste der gefundenen Dokumente ausgegeben. Liefert die Suche kein Dokument zurück, wird eine entsprechende Meldung ausgegeben. Hat der Benutzer falsche Angaben im Formular gemacht wird eine Fehlermeldung mit einem genauen Hinweis auf den Fehler ausgegeben.

5.6.2.5 Dokumente einspielen

Servletname: SportDL_Einspielen, SportDL_Dokument_einspielen

Benutzte Klassen und Pakete: java.sql, java.io, javax.servlet, javax.servlet.http, java.util, com.oreilly.servlet.MultipartRequest
SportDL_ODBC, SportDL_FCDB, SportDL_Sitepool

Grundsätzliche Arbeitsweise: Die Funktion *Dokumente Einspielen* ist zweistufig implementiert. In der ersten Stufe muss der Nutzer die Dokumentart bestimmen, zu der er ein Dokument einspielen möchte. Das Servlet *SportDL_Einspielen* regelt dann den Aufbau des für die gewählte Dokumentart zu verwendenden Html-Formulars.
Für den Fall, dass der Nutzer ein Abstract zu einem bereits im e-Journal eingestellten Volltext einspielen möchte, übernimmt dieses Servlet auch die Suche und die Referenzierung des Volltextes, um somit in der Dokumentdatenbank eine Verknüpfung zwischen Abstract und Volltext zu ermöglichen.
In der zweiten Stufe werden die vom Nutzer angegebenen Daten und die Datei zu dem Dokument auf ihre Konsistenz geprüft und in der Dokumentdatenbank bzw. dem Dokument-Verzeichnis gespeichert. Das Servlet *SportDL_Dokument_einspielen* stellt diese Funktionen zur Verfügung.

Festlegen der Dokumentart (SportDL_einspielen) Als Parameter erhält das Servlet per Servlet-Request das Login den Nutzers und ein Zeichen zur Unterscheidung der anzusprechenden Funktion.
Für die Fälle *Volltext einspielen* und *sonstige Dokumente einspielen* wird lediglich aus dem Sitepool-Objekt das entsprechende Html-Formular generiert und per Servlet-Response an den Client des Nutzers geschickt.
Soll ein Abstract zu einem im e-Journal bereits eingestellten Volltext eingespielt werden, wird der Titel des Volltextes abgefragt und über die Datenbank-Schnittstelle werden alle zu dem Suchbegriff passenden Titel aus der Dokumentdatenbank gesucht. Die Daten aus dem Result-Set werden ausgelesen und an das Sitepool-Objekt übergeben, welches den Html-Code mit den Suchergebnissen generiert, der dann an den Client des Nutzers gesendet wird. Zu jedem gefundenen Dokument wird die Dokumentnummer und der Titel in die Parameterliste der URL aufgenommen. Der Titel und die Dokumentnummer werden ausgelesen, wenn das Servlet mit einem bestimmten Parameter aufgerufen wird. Das Servlet generiert dann ein Html-Formular zum Einstellen des Abstracts zu dem vorher gefundenen Volltext.

Einstellen des Dokuments (SportDL_Dokument_einspielen) Als Parameter erhält das Servlet per Servlet-Request das Login den Nutzers und einen Parameter zur Unterscheidung des Typs des einzustellenden Dokuments. Soll ein Abstract zu einem Volltext eingestellt werden, wird außerdem noch die Dokument-Nummer des Volltextes, den der Nutzer zuvor im e-Journal gefunden hat, an den Server übermittelt.
Per Servlet-Request werden die in das Html-Formular eingetragenen Metadaten ausgelesen und es wird geprüft, ob die Daten vollständig und korrekt sind. Ist dies der Fall, werden die Daten in die Dokumentdatenbank eingetragen und es wird per Multipart-Request die Datei an den Server übertragen und in das Dokumentverzeichnis unter der Dokument-Nummer gepeichert. Zuvor wird geprüft, ob die Datei ein Word- bzw. PDF-Dokument ist. Dateien mit anderen Formaten werden vom System abgewiesen und der Nutzer wird aufgefordert, das richtige Format zu verwenden. Wenn alle Daten und das Dateiformat korrekt sind erhältDer Nutzer zur Bestätigung eine Meldung über den Erfolg der Einstellung.

5.6.2.6 Ändern der Benutzerdaten

Servletname: SportDL_Konfiguration

Benutzte Klassen und Pakete: java.sql, java.io, javax.servlet, javax.servlet.http, SportDL_ODBC, SportDL_Sitepool

Grundsätzliche Arbeitsweise: Ähnlich wie bei der Registration werden die eingegebenen Daten zunächst auf Korrektheit überprüft und dann in die Oracle-Datenbank eingetragen. Auch hier gilt, dass die Email-Adresse das at-Zeichen '@' enthalten muss. Unabhängig von den Benutzerdaten kann auch das Passwort geändert werden. Hierzu werden die Strings, die in die beiden Felder eingegeben wurden auf Gleichheit überprüft und bei Übereinstimmung abgespeichert. Stimmen die Passwörter nicht überein, so wird das alte Passwort beibehalten. Bei korrekter Eingabe der Daten wird eine HTML-Seite aufgebaut, die den Nutzer darüber informiert, dass seine Daten abgeändert wurden. Ansonsten erscheint eine entsprechende Fehlermeldung.

Probleme bei der Implementation: Auch hier gilt darauf zu achten, dass bei der Zusammensetzung der SQL-Statements jedes einzelne Datum in Hochkommata zu setzen ist. Ansonsten werden die Daten nicht korrekt abgespeichert.

5.6.2.7 Kommunikationsmöglichkeiten mit Mailinglisten News

Servletname: SportDL_Kommunikation

Benutzte Klassen und Pakete: javax.servlet, javax.servlet.http, java.io, java.sql

Grundsätzliche Arbeitsweise: Die Klasse SportDL_Kommunikation stellt den Zugang zum Hypernews-System und zu den Mailinglisten dar. Hypernews wird über einfach Links angebunden, da die Newsseiten in eigenen Browser-Fenstern dargestellt werden. Bei den Mailinglisten ist es möglich, sich per Klick auf Checkboxen an- oder abzumelden. Der aktuelle Status der Anmeldungen wird beim Aufbau der Seite aus der Oracle-Datenbank gelesen. An- oder Abmeldungen werden in Form einer E-Mail mit java-mail an das Mailinglisten-System MajorDomo verschickt und die Änderungen wieder in der Datenbank aktualisiert.

Konzeptionelle Fehler: MajorDomo verschickt von sich aus eine Mail an den Benutzer, wie er sich wieder abmelden kann. Sollte er das wirklich selber per E-Mail tun, so wird diese Information in der Datenbank ungültig, weil die Datenbank nicht aktualisiert wird. Falls der Benutzer sich nachträglich wieder über die SportDL bei MajorDomo anmelden möchte, so wird das nicht geschehen, weil nach der Datenbankabfrage keine neue Anmeldung abgeschickt wird. Der Benutzer muß sich also erst über die SportDL abmelden und dann wieder anmelden, bis er wieder in die Mailingliste aufgeneommen wird. Für dieses Problem müßte festgestellt werden, ob man MajorDomo so konfiguriren kann, daß er diese Informationen nicht mehr verschickt bzw. die Informationen abändert, die verschickt werden.

5.6.3 Beschreibung der Hilfsklassen

5.6.3.1 SportDL_ODBC

Benutzte Klassen und Pakete: java.sql, java.io

Funktionalität: Diese Klasse dient lediglich zur Kapselung der Daten, die zum Anmelden an der Oracle-Datenbank notwendig sind. Jegliche Statements die diese Klasse erhält, werden unverändert an die Java-SQL-Schnittstelle weitergegeben. Ebenso gibt sie die Resultsets nicht modifiziert zurück. Die Verbindung zur Datenbank wird aufgebaut, sobald ein Objekt dieser Klasse instantiiert wird. Entsprechend wird die Verbindung bei Zerstörung des Objekts getrennt.

5.6.3.2 SportDL_Mail

Benutzte Klassen und Pakete: java.util, javax.mail, javax.mail.internet, SportDL_Sitepool

Funktionalität: Die Klasse SportDL_Mail baut auf dem JavaMail-Paket von SunSoft auf und vereinfacht das Abschicken von Emails per Servlet. Es müssen nur Empfänger, Absender, Thema der Email und der Mailtext im Konstruktor angegeben werden und schon wird die Mail nach dem Instantiieren abgeschickt.

Probleme bei der Implementation: Es ist wichtig, dass die Adresse des Absenders eine gültige Email-Adresse ist. Ansonsten wird die Email vom Mailserver eventuell nicht korrekt weitergeleitet.

5.6.3.3 SportDL_FCDB

Benutzte Klassen und Pakete: eVerlage

Funktionalität: Die Klasse SportDL_FCDB baut auf dem dem Paket eVerlage des geleichnamigen Offis-Projekts auf und dient zur Kapselung der Vorgänge, die für die Kommunikation mit Fulcrum notwendig sind. Es werden anwendungsspezifische Methoden (in Bezug auf die SportDL) zur Verfügung gestellt, um PDF-Dokumente in die Fulcrum-Datenbank einzustellen und gleichzeitig Fulcrum dazu zu veranlassen diese zu indexieren, sowie Dokumente wieder zu entfernen. Außerdem existiert eine Methode für die Volltextsuche in einem ausgewählten Dokument.
Diese Methoden werden über eine Objekt der Klasse SportDL_FCDB aufgerufen, das bereits bei der Instantiirung eine Verbindung zu Fulcrum hergestellt. Das Objekt kapselt für den Nutzer alle Informationen, die die Verwaltung der Dokumente mit Fulcrum betreffen, wie z.B. Tabellennamen oder die Struktur von SQL-Satements, über die die eigentliche Kommunikation mit der Datenbank stattfindet. Um die Methoden dieser Klasse zu nutzen, müssen lediglich die Namen Dokumente, die verarbeitet werden sollen, bekannt sein.

Probleme bei der Implementation: Beim Zugriff auf Fulcrum über Servlets bzw. eine Java Virtual Machine ist es notwendig die erforderlichen Umgebungsvariablen von Fulcrum zu setzen. Unter *apache* stehen diese Variablen in einem Abschnitt in der Datei *jserv.properties*.

5.6.3.4 SportDL_Sitepool

Benutzte Klassen und Pakete: java.io

Funktionalität: Die Klasse SportDL_Sitepool stellt Methoden zur Verfügung, die HTML-Code in der Form von Strings zurückliefern. Sie wird von allen anderen Klassen der SportDL mit den entsprechenden Parametern aufgerufen, welche auch Strings sind, um dann die HTTP-Ausgaben für die Benutzerschnittstelle zu realisieren. Das ursprünglich Ziel war, alle HTML-Ausgaben aus dieser Klasse heraus zu produzieren. Dies erschien sinnvoll, weil die SportDL aus mehreren Servlets besteht, die alle ähnliche HTML-Ausgaben durchführen müssen, und man so eine Klasse zur Verfügung hätte, die den HTML-Code in Form von selbstgestalteten Templates vorrätig hätte.
Allerdings wurde während der Implementationsphase festgestgestellt, daß einige Ausgaben einfacher zu codieren sind, wenn sie direkt in den anderen Klassen implementiert werden, weswegen das Konzept aufgeweicht wurde, was die Wartung der Benutzerschnittstelle erschweren wird. Über ein neues Konzept wurde schon nachgedacht, allerdings reichte die Zeit zur Implementierung für den Prototyp nicht mehr aus.

5.7 Handbuch

5.7.1 Suche und eJournal

Mit dieser einfachen Suche ist es möglich, die SportDL nach Autor oder Titel zu durchsuchen. Die Eingabe von sogenannten Wildcards ist erlaubt. So steht das Prozentzeichen % für eine beliebige Zeichenkette. Beispiel: Die Suchanfrage nach dem Autor „K%er“ liefert sowohl Küster, wie auch Kederer oder Ker. Falls die Suche verfeinert werden soll, bringt ein Klick auf „erweiterte Suche“ sie zu einer Seite, auf der sogar eine Volltextsuche über die Dokumente der Sport-DL angeboten wird.

Die vier weiteren Links mit Namen „eJournal“, „Abstracts“, „Forschungsnotizen“ und „Sonstiges“ führen zu einer vollständigen, alphabetischen Auflistung der Dokumente des entsprechenden Typs.

5.7.2 Suchergebnisse

Auf dieser Seite können sie das Ergebnis ihre Suchanfrage begutachten. Je nach Typ des angezeigten Dokuments haben sie verschiedene Möglichkeiten:

5.7.2.1 Volltext

Klicken sie auf den Titel und das Dokument wird im Browser dargestellt. Sollte sich nun ein Fenster mit dem Titel „Save as ...“ öffnen, dann haben sie das Acrobat-Reader-Plugin noch nicht installiert. Unter www.acrobat.com läßt sich das Plugin kostenlos herunterladen. Ein Klick auf den Download-Button ermöglicht das Speichern des Dokumentes auf ihrer Festplatte. Klicken sie auf „Abstracts anzeigen“ und es wird eine Liste aller zu diesem Dokument vorhandenen Zusammenfassungen angezeigt. Der Link „Literaturliste anzeigen“ stellt die zu diesem Dokument vorhandene Literaturliste in ihrem Browser dar.

5.7.2.2 Abstract, Forschungsnotizen und Sonstiges

Klicken sie auf den Titel und das Abstract wird im Browser dargestellt. Sollte sich nun ein Fenster mit dem Titel „Save as ...“ öffnen, dann haben sie das Acrobat-Reader-Plugin noch nicht installiert. Unter www.acrobat.com läßt sich das Plugin kostenlos herunterladen. Ein Klick auf den Download-Button ermöglicht das Speichern des Dokumentes auf ihrer Festplatte.

5.7.3 Konfiguration ändern

Hier können sie ihre persönlichen Benutzerdaten ändern. Sind sie mit den eingegebenen Daten zufrieden, dann klicken sie auf den Button 'Abschicken' und ihre Angaben werden in der Datenbank gespeichert. Zum Ändern des Paßworts müssen sie dieses zweimal eingeben, wobei es kein Minimum für die Länge des Paßwortes gibt.

5.7.4 Registration

Sie müssen sich zunächst registrieren, um die Seiten der SportDL nutzen zu können. Geben sie dazu ihren gewünschten Benutzernamen und ihre Email-Adresse an. Bedenken Sie, daß der Benutzername mindestens 5 Zeichen enthalten muß. Alle anderen Angaben sind freiwillig. Klicken sie dann auf 'Abschicken' und sie erhalten umgehend eine Email, die ihren Benutzernamen und ihr Paßwort beinhaltet. Jetzt können sie sich bei der SportDL anmelden.

5.7.5 Die Erweiterte Suche

Das Formular zur Erweiterten Suche bietet vielfältige Möglichkeiten für die Recherche nach Dokumenten innerhalb der SportDL. Es kann in erster Linie nach drei unterschiedlichen Typen von Informationen gesucht werde: Dem Titel eines Dokuments, dem oder den Autoren, oder nach Begriffen, die in einem Dokument auftreten sollen. Es müssen allerdings nicht alle Felder ausgefüllt werden. Welche Art von Information in einem der drei Textfelder steht kann über die Auswahlfelder davor frei eingestellt werden. Als logische Verknüpfungen zwischen den Feldern stehen `UND` und `ODER` zur Auswahl. Über das weitere Kriterium Veröffentlichungsjahr, das als vierstellige Zahl eingegeben werden kann, hat der Nutzer die Möglichkeit, die Suche noch weiter einzuschränken. Mit der Option `'frei'` bleibt dieses Kriterium unberücksichtigt. Abschließend muss nur noch über die Kontrollkästchen am unteren Ende festgelegt werden auf welchen Dokumenttyp die Suche eingeschränkt werden soll. Voreingestellt ist anfangs nur das `eJournal`, es dürfen aber alle Typen frei kombiniert werden. Ist kein Kontrollkästchen aktiviert, wird automatisch nur im eJournal gesucht. Ein Mausklick auf die Schaltfläche `Suchen` startet den Suchprozess.

Hinweis: Volltextangaben dominieren bei der Suche z.Zt. über die Angaben zum Autor oder Titel eines Dokuments. Dies bedeutet, dass zunächst alle Dokumente, die zum Autor oder Titel passen, gesucht werden und dann wird erst geprüft (unabhängig von der eingestellten Verknüpfung), ob die Volltextangabe(n) ebenfalls zu diesem Ergebnis passen. Nur wenn dies zutrifft erscheint das Dokument im Suchergebnis.

5.7.6 Dokumente einspielen

Unter Dokumente einstellen ist das Einstellen von Volltexten, Forschungsnotizen und Abstracts zu Volltexten, die bereits in das e-journal eingestellt wurden, möglich.
Zunächst mßsen Sie wählen, welche Dokumentart Sie einstellen möchten.

5.7.6.1 Volltext einspielen

Geben Sie bitte den Titel, den Autor oder die Autoren (durch ; getrennt), das Veröffentlichungsjahr sowie die Sprache an. Dies sind Angaben, die Sie machen müssen. Optional können Sie noch eine kurze Beschreibung und ein Thema zu Ihrem Dokument angeben. Im Feld 'Volltext' müssen Sie den Pfad zu Ihrer Dokumentdatei angeben. Außerdem muss im Feld 'Abstract' eine einseitige Zusammenfassung zu dem Volltext mit eingespielt werden. Das Einspielen einer Literaturliste ist optional. Dabei ist zu beachten, dass sich die Dateinamen in mindestens einem Zeichen unterscheiden müssen.

5.7.6.2 Abstract zu einem Volltext aus dem e-Journal einspielen

Zunächst müssen Sie den Titel des Volltextes zu dem Sie ein Abstract einspielen wollen in das Suchfeld eintragen und auf die Schaltfläche 'Suchen' klicken. Sie Bekommen dann die e-Journalbeiträge angezeigt, die den von Ihnen eingegeben Titel haben oder beinhalten. Durch Anklicken des entsprechenden 'Abstract einspielen' Buttons bekommen Sie das Einspielenformular angezeigt. Geben Sie bitte den Titel, den Autor oder die Autoren (durch ; getrennt), das Veröffentlichungsjahr sowie die Sprache an. Dies sind Angaben, die Sie machen müssen. Optional können Sie noch eine kurze Beschreibung und ein Thema zu Ihrem Dokument angeben. Im Feld 'Abstract zum Volltextdokument' müssen Sie den Pfad zu Ihrer Dokumentdatei angeben.

5.7.6.3 Sonstige Dokumente einspielen

Auf dieser Seite können einzelne Abstracts zu anderweitig publizierten Dokumenten , Forschungsnotizen und sonstige Dokumenttypen in die SportDL eingestellt werden. Geben Sie bitte den Titel, den Autor oder die Autoren (durch ; getrennt), das Veröffentlichungsjahr sowie die Sprache an. Dies sind Angaben, die Sie machen müssen. Optional können Sie noch eine kurze Beschreibung und ein Thema zu Ihrem Dokument angeben. Im Feld 'Dokumenttyp' muss angegeben werden, welche Art Dokumnt eingestellt werden soll. Dahinter müssen Sie den Pfad zu Ihrer Dokumentdatei angeben.

Wichtig: Es ist zu beachten, dass nur Dateien mit der Endung .PDF oder .DOC angenommen werden können. Wenn alle Angaben vollständig eingetragen sind werden sie durch anklicken der Schaltfläche 'Abschicken' an den Server der SportDL übertragen und Sie erhalten eine Meldung über den erfolgreichen Abschluss der Übertragung.

5.7.7 Kommunikation

Die SportDL stellt Ihnen den Zugriff auf die SportDL-Newsgroups und SportDL-Mailinglisten zur Verfügung.

5.7.7.1 Die SportDL-Newsgroups

In einer Newsgroup kann man über die unterschiedlichsten Themen und Probleme diskutieren, man findet oft Rat und Antwort in Ihnen. Man schreibt an eine Newsgroup entweder zu einem eigenen Thema oder äußert sich zu bestehenden Themen oder Fragen. Bei der SportDl existieren zur Zeit die Newsgroups „Neues von der SportDL“ und „Anmerkungen zu den eingestellten Dokumenten“.

5.7.7.2 Benutzung der SportDL-Newsgroups ?

Klicken Sie auf den Link der gewünschten Newsgroup. In dem Fenster, welches neu geöffnet wird, finden Sie Links, die zu bestehenden Fragen und Antworten füren. Sie können entweder diesen Links folgen, die schon bestehenden Beiträge lesen und bezogen auf diese Beiträge antworten,

oder Sie klicken auf den Knopf „Add a Message“, welcher Ihnen ein Formular öffnet, in welches Sie Ihren Beitrag schreiben können.

Folgen Sie den Links zu bestehenden Beiträgen, so können Sie in den erscheinenden Beiträgen den Knopf Ädd a messageänklicken, um bezogen auf die schon bestehenden Beiträge zu antworten.

5.7.7.3 NEWSGROUP : Neues von der SportDL

In dieser Newsgroup können generelle Anmerkungen zur SportDL gemacht werden. Dazu gehören u.a. Termine, Informationen zur SportDL selber, Fragen und Antworten zur SportDL, Vorschläge zur SportDL oder zu gewünschten Dokumenten usw.

5.7.7.4 NEWSGROUP : Anmerkungen zu den eingestellten Dokumenten

Hier können die Nutzer der SportDL sich über die Dokumente in der SportDL austauschen, Fragen zu den Dokumenten stellen, sich diese Fragen gegenseitig beantworten und darüber debatieren.

5.7.7.5 Die SportDL-Mailinglisten

Wenn Sie sich bei der SportDL auf einer Mailingliste eintragen, so bekommen Sie in unregelmäßigen Abständen Informationen zu dem Thema der Mailingliste per E-Mail zugeschickt. Diese Informationen werden bei der SportDL meistens Informationen zu Neuerscheinungen von Dokumenten in der SportDL sein. Die SportDL bietet die folgenden Mailinglisten an :

- eJournal:
  Hier bekommen Sie Informationen zu neu eingestellten Volltexten zugeschickt.
- Abstracts:
  Hier bekommen Sie Informationen zu neu eingestellten Abstracts zu Volltexten zugeschickt.
- Forschungsnotizen:
  Hier bekommen Sie Informationen zu neu eingestellten Forschungsnotizen zugeschickt.
- Sonstiges:
  Hier bekommen Sie Informationen zu sonstigen Dokumenten oder Ereignissen bzgl. der SportDL zugeschickt.

5.7.7.6 Registration bei den Mailinglisten der SportDL

Auf der Seite Kommunikation sind links neben vier Mailinglisten eJournal, Abstracts, Forschungsnotizen und Sonstiges Checkboxen zu sehen. Ist in so einer Checkbox ein Haken, so sind Sie bei dieser Mailingliste angemeldet, ist der Haken nicht da und das Feld ist grau, so sind sie dort noch nicht angemeldet.
Wenn Sie sich bei einer oder mehreren Mailinglisten anmelden wollen, so klicken Sie die

entsprechenden Checkboxen an. Haben Sie nun alle Mailinglisten ausgewählt, von denen Sie Informationen bekommen wollen, so klicken Sie auf den Knopf „Abschicken“ am unteren Teil der Seite.
Sind Sie auf einer Mailingliste eingetragen und Sie wollen sich bei dieser oder bei mehreren Mailinglisten wieder austragen, so klicken Sie auf die entsprechenden Checkboxen, so daß der Haken in der entsprechenden Box verschwindet. Haben Sie sich bei allen gewünschten Mailinglisten ausgetragen, so klicken Sie auf den Knopf Äbschickenäm unteren Teil der Seite.

WICHTIG :

Sie bekommen von den Mailinglisten, auf denen Sie eingetragen sind, Hinweise darauf, wie sie sich per E-Mail wieder austragen können. Ignorieren Sie bitte diese Hinweise, denn Sie dürfen sich nicht per E-Mail aus diesen Mailinglisten austragen! Der Grund ist, daß die SportDL in einer Datenbank gespeichert hat, ob Sie bei welchen SportDL-Mailinglisten Sie eingetragen sind und bei welchen nicht. Die SportDL wird auch dann noch annehmen, daß Sie auf einer Mailingliste eingetragen sind, auch wenn Sie sich schon per E-Mail selber abgemeldet haben. Wollen Sie sich dann später erneut auf dieser Mailingliste wieder eintragen, so wird die SportDL dies nicht tun, weil die SportDL davon ausgeht, daß Sie noch angemeldet sind. Für diesen Fall tragen Sie sich bei der SportDL bei dieser Mailingliste wieder aus und dann wieder ein, um die Datenbank auf den richtigen Stand zu bringen.

Index

Abstract Interface Design, 136, 140
Access Structures, 139
ADV, 70, 140
Animation, 80
Attributbelegungen, 85
    extern, 85, 86
    intern, 85, 86
    Normalwert, 86
Attribute, 81
    Übelkeit, 85, 93
    Angstgefühl, 83, 93
    Atemfrequenz, 83, 86, 87
    Atemgeräusche, 83, 87
    Atemtiefe, 83, 86, 87
    Außentemperatur, 93
    Ausentemperatur, 83
    Bekleidung, 83, 93
    Bewußtsein, 82, 83, 93
    Blutdruck, 83, 86
    Blutungsart, 83, 91, 92
    Blutungsintensität, 83, 87–92
    Blutvolumen, 83, 86
    Durst, 83, 93
    Farbe, 83, 87–92
    Fremdkörper, 84, 87–92
    Gefahrenquelle, 81, 84, 93
    Herzrhythmus, 84, 86
    Herzschlag, 84, 86
    Hilfsmittel, 84, 93
    Kohlenmonoxidvergiftung, 84, 86
    Kommunikation, 84, 93
    Krankheit, 84, 93
    Lage der Extremitäten, 84, 93
    Lageart, 84, 93
    Lageort, 84, 93
    Lokalisation, 84, 89
    Nervosität, 84, 93
    Puls am Hals, 84, 86
    Puls am Handgelenk, 84, 86
    Pupillengröße, 84, 91
    Sauerstoffgehalt des Blutes, 84, 86
    Schmerzcharakteristik, 85, 87–92
    Schmerzintensität, 85, 87–92
    Schock, 85, 86
    Schweißbildung, 85, 90
    Schwindel, 85, 93
    Sehstörung, 84, 91
    Speichelfluß, 85, 92
    Subjektives Temperaturempfinden, 85, 93
    Temperatur, 85, 87–92
    Tränenfluß, 85, 91
    Verletzung, 85, 87–92
    Wetterlage, 85, 93
    Zeit, 85, 93
Attributgruppen
    Äußere Umstände, 82, 83, 93
    Auge, 82, 91
    Bauchraum, 82, 87
    Becken, 82, 88
    Brustkorb, 82, 88
    Extremitäten, 82, 89
    Gesicht, 82, 90
    Herz-Kreislaufsystem, 82, 86
    Kopf, 82, 90
    Lunge, 82, 87
    Mund, 82, 92
    Nase, 82, 83, 92
    Ohrmuschel, 82, 83, 91
    Opfer, 82, 83, 93
    Rücken, 82, 89
    Rachen, 82, 87
    Rumpf, 82, 88

Bedienung, 79
Benutzungsschnittstelle, 79
Bildschirmausgaben, 79
Button, 67

Conceptual Design, 136, 137

Fallbeispiel, 79

Gesamtkurs, 61
Graphik, 80
Grouping, 54
Guided Tour, 54

Hilfe, 81
Hilfesystem, 69
History-Funktion, 69

Index, 54
Index Guided Tour, 54
Indexsuche, 67
Interaktion, 79

Layout, 67
Lektionen, 54, 80
Links, 139

Nachschlagewerk, 61
Navigation, 67
Navigational Design, 136, 138
Navigationsgraphen, 53, 60
Navigationsnotation, 53
Nodes, 138
Notfallmasnahmen, 79

Objektgraph, 60
OOHDM, 136
Organisatorisches, 65

Simulation, 79
Situationsbeschreibung, 79
Symptome, 64, 79

Text, 80

Unfallhergang, 79